APTD - 1153

THE COMBINED FILE SEARCH SYSTEM

PROGRAM MANUAL

FOR

PB 211 070

AN INFORMATION STORAGE

AND

RETRIEVAL SYSTEM

PREPARED FOR

CONSUMER PROTECTION AND

ENVIRONMENTAL HEALTH SERVICE

BY

THE SERVICE BUREAU CORPORATION

Reproduced by
NATIONAL TECHNICAL
INFORMATION SERVICE
U S Department of Commerce
Springfield VA 22151

JULY 1968

315

BIBLIOGRAPHIC DATA SHEET

1. Report No.
APTD-1153

2.

3. Recipient's Accession No.
PB-211 070

4. Title and Subtitle
The Combined File Search System - Program Manual for an Information Storage and Retrieval System

5. Report Date
July 1968

6.

7. Author(s)

8. Performing Organization Rept. No.

9. Performing Organization Name and Address
The Service Bureau Corporation

10. Project/Task/Work Unit No.

11. Contract/Grant No.

12. Sponsoring Organization Name and Address
Consumer Protection and Environmental Health Service
The Food and Drug Administration
Washington, D. C.

13. Type of Report & Period Covered

14.

15. Supplementary Notes

16. Abstracts

The technical approach taken in the solution of the problems encountered in developing an information storage and retrieval system is defined. Since many of the programs were common to a number of separate files, macros and generalized routines were written which are used by many programs. The design and usage of these macros and routines are discussed in detail. In addition to the generalized routines, the major programs are described in detail. Since this manual intends to define the detailed logic of the system, each program is treated as a separate entity.

17. Key Words and Document Analysis. 17a. Descriptors

Information retrieval
Information systems
Computer programs
Manuals

17b. Identifiers/Open-Ended Terms

17c. COSATI Field/Group 09 02, 05 02

18. Availability Statement
Unlimited

19. Security Class (This Report)
UNCLASSIFIED

20. Security Class (This Page
UNCLASSIFIED

21. No. of Pages
315

22. Price
$6.00

FORM NTIS-35 (REV. 3-72)

USCOMM-DC 14952-P72

THE COMBINED FILE SEARCH SYSTEM

developed for

THE FOOD AND DRUG ADMINISTRATION

by

THE SERVICE BUREAU CORPORATION

PROGRAM MANUAL

TABLE OF CONTENTS

INTRODUCTION 1

MACRO USAGE 2

GENERALIZED ROUTINES 9

- DADIO ROUTINE 9
- PLINE ROUTINE 10
- XSORT ROUTINE 11
- LKDICT ROUTINE 12

DICTIONARY PROGRAMS 20

- TABBLD PROGRAM 20
- DEDIT PROGRAM 29
- DICT2 PROGRAM 74
- DKDICT ROUTINE FLOWCHARTS 85
- DICTMSG PROGRAM 87
- DPRINT3 PROGRAM 90
- SBRFORM ROUTINE 102
- SYNTAPE PROGRAM 110
- DPRINT4 PROGRAM 113
- RELTERM PROGRAM 133
- RELTRM1 PROGRAM 136

MAIN FILE PROGRAMS 140

- MFEDIT PROGRAM 140
- MFMAINT PROGRAM 188
- LKXREF ROUTINE FLOWCHARTS 215
- IVUPDT PROGRAM 216
- IFPRNT PROGRAM 224
- IVPRINT PROGRAM 228
- UPDXREF PROGRAM 233
- XREFPRNT PROGRAM 238
- IDUPDATE PROGRAM 243
- MPRINT1 PROGRAM 246
- MPRINT2 PROGRAM 256
- DUMPCELL PROGRAM 257
- LOADCELL PROGRAM 270

Table of Contents (continued)

SEARCH PROGRAM 273

APPENDIX A - DICTIONARY RECORD 306

APPENDIX B - MASTER FILE RECORD 307

APPENDIX C - INVERTED FILE RECORD 309

APPENDIX D - CROSS REFERENCE FILE RECORD 310

APPENDIX E - SEARCH TAPE OUTPUT RECORD 311

INTRODUCTION

It is the intent of this program manual to define the technical approach taken in the solution of the problems encountered in developing an information storage and retrieval system. Since many of the programs were common to a number of separate files, macros and generalized routines were written which are used by many programs. The design and usage of these macros and routines are discussed in detail in the following sections. In addition to the generalized routines, the major programs are described in detail within the system in which they function. That is, a dictionary print program will be described within the dictionary system section. However, since the generalized routines may be used by programs in all three sections, they are described separately.

This modular approach allows the user a certain flexibility in adapting the system to his own needs. For example, if a disk were to be used for storage of the master file instead of a data cell, the only change necessary would be the alteration of the parameters of the output macro.

Since this manual intends to define the detailed logic of the system, each program is treated as a separate entity. For a more general description of the system and the interaction of the files, please refer to the User's Manual.

MACRO USAGE

The following macro definitions are used the the Food and Drug Administration Information Storage and Retrieval System programs to standardize and simplify some coding sequences. If the macro definitions are not cataloged, those used must be included with the source deck at assembly time. A typical source deck contains the JOB card, the EXEC ASSEMBLY card, the macro definitions, and the source deck in that order.

Disk/Data Cell Input/Output

The Direct Access Device Input/Output (DADIO) routine transfers data between main storage and the 2311 disk file or the 2321 data cell. The following macro definitions generate the linkage between a calling program and the DADIO program.

OPERATION CODE	OPERANDS	OPERATIONS
RDSK	SYSXXX,A2,IOA	Read Disk
WDSK	SYSXXX,A2,IOA	Write Disk
RDCL	SYSXXX,A3,IOA	Read Data Cell
WDCL	SYSXXX,A3,IOA	Write Data Cell

Description of Operands

1. SYSXXX

   Enter the symbolic unit (SYS000 - SYS244)

   a. SYS010 is the dictionary disk

   b. SYS014 is inverted file disk

   c. SYS018 is the cross reference file disk

d. SYS020 is the work file disk for search

e. SYS031 is the master file on data cell

2. A2

Enter the address of a two byte field containing a binary disk track address. DADIO converts this address to a cylinder head disk address.

3. A3

Enter the address of a three byte field containing a binary data cell track address. DADIO converts this address to the data cell track address (cell-subcell-strip-cylinder-head).

4. IOA

Address of the input/output area. This must include an eight byte count area which is used by the DADIO routine. This count area must immediately precede the I/O area. To write on disk, the first two bytes after the count area must contain the length of the record to be written on the data cell. This length is the length of the data record on disk or data cell and does not include the eight byte count area.

Direct Access Look Up

The following macro definitions generate the linkage between a calling program and the desired look up programs.

OPERATION CODE	OPERANDS	OPERATION
LKDF	TM,OUT,ARTN	Look Up Dictionary File
LKIF	TM,OUT,ARTN	Look Up Inverted File Record
LKXF	XDN,OUT,ARTN	Look Up Cross Reference File Record
LKMF	IDN,OUT,ARTN	Look Up Master File Record

Description of Operands

1. TM

   Enter the address of the term or descriptor to be searched for in the disk file. The first byte of the term contains the length of the descriptor plus one followed by a descriptor of up to 64 characters.

2. XDN

   Enter the address of an eight byte external document number.

3. IDN

   Enter the address of a three byte internal document number and one byte segment number.

4. OUT

   Enter the address of the output area where the record (if found) will be put.

5. ARTN

   Enter an alternate return address to be taken if the record cannot be found. If this operand is omitted, the next sequential instruction will be executed for a record which is not found.

   NOTE: If the cross reference look up program cannot find the external document, it is added to an overflow area on the disk and the alternate return address is taken.

6. OPTN

   This option is usually blank. Enter 'TEXT' to retrieve the next text segment in the master file. Enter 'ALL' to retrieve the next sequential document in the master file.

Put Indexed Sequential

An indexed sequential disk file is created for the dictionary, the inverted file, and the cross reference file as each of these tape files is updated. The following specialized macro definitions generate the linkage between the tape update programs and the put indexed sequential routines.

OPERATION CODE	OPERANDS	OPERATION
DKDCT	OP,REC	Put dictionary disk (DKDICT)
DKIVT	OP,REC	Put inverted file disk (DKIVRT)
DKRF	OP,REC	Put cross reference file disk (DKXREF)

Description of Operands

1. OP

   Enter an operation code as follows:

   a. 0 = open, initialize index tables

   b. 1 = close, write final index tables

   c. 2 = output record and update indexes

2. REC

   Address of record to output. This operand is unnecessary for OP = 0 or 1.

Miscellaneous Macros

OPERATION CODE	OPERANDS	OPERATION
PRNT	MSG,LEN	Print a line on the printer (PLINE)

Description of Operands

1. MSG

   Address of line to print

2. LEN

   Length of print line; if 0, skip to channel 1

OPERATION CODE	OPERANDS	OPERATION
TYPE	MSG,LEN,RTY	Type a message to operator. He must reply with CANCEL, IGNORE, or RETRY. (PLINE)

Description of Operands

1. MSG

   Address of message to print

2. LEN

   Length of message to print

3. RTY

   Branch address if operator enters RETRY. IGNORE takes the next sequential instruction. CANCEL cancels the job.

OPERATION CODE	OPERANDS	OPERATION
XDMP	START,LEN,CNT	Debugging core dump routine

Description of Operands

1. START

   Start address of core printout

2. LEN

   Number of bytes to print in dump

3. CNT

   Number of times to print this area (CNT = 1 to 254)

The program XDUMP will only dump when UPSI 4 is on (// UPSI XXXX1XXX).

OPERATION CODE	OPERANDS	OPERATION
PRNTLC	ADDR,CNT	Initialize count of lines per page (PLINE)

Description of Operands

1. ADDR

   Address of user's page heading routine

2. CNT

   Count of lines per page on printer

OPERATION CODE	OPERANDS	OPERATION
XSRT	ACNT,AREA,LEN, SBEG,SLEN	Generalized sort routine linkage

Description of Operands

1. ACNT

   Address of half-word count of terms to sort

2. AREA

   Address of buffer of terms to sort

3. LEN

   Length of each record to sort

4. SBEG

   Beginning of sort field

5. SLEN

   Length of sort field

GENERALIZED ROUTINES

1. Direct Access Devices Input/Output Routine (DADIO)

Processing

This routine is used for all data transfers between the disk and data cell files in this system and can be used for disk work files. Linkage to this routine is generated by using the RDSK, WDSK, RDCL, or WDCL macros. The first time each file is accessed, that file is opened using the DOS OPEN routines. During OPEN, the extent information is stored in a table of opened files. There is one entry in this table for each file used in the current program being executed. DADIO constructs a CCW chain for the read or write operation and also computes the seek address from a binary number. The routine base register (15) and the return address register (14) are both loaded by the user.

Record Formats

a. Input linkage from user-register 1

POSITION	LENGTH	FIELD DESCRIPTION
0	1 byte	Logical unit number
1 - 3	3 bytes	Address of binary track address
4	1 byte	Switch (X'00'-RDSK, X'20'-WDSK,X'40'-RDCL,X'60'-WDCL)
5 - 7	3 bytes	Address of I/O area (includes eight byte count areas)

b. Table of opened files

POSITION	LENGTH	FIELD DESCRIPTION
0	1 byte	Logical unit number
1	1 byte	Number of volumes for this file
2	1 byte	Switch (X'80'-INPUT, X'40'-OUTPUT,X'20'-DISK,X'10'-DATA CELL)
3	1 byte	Unused
4 - 5	2 bytes	Starting track of volume (disk only)
6 - 7	2 bytes	Number of tracks (disk only)

2. Printer and Console Output Routine (PLINE)

Processing

This routine has three entry points: 1) print a line; 2) type a line and wait for reply from operator; and, 3) initialize line control feature. To print a line the linkage from the caller contains a length of print line and an address to print from. Both of these fields are inserted into the CCW for the printer. The operation to type a line is the same as printing except that the operator must enter RETRY, IGNORE, or CANCEL before control will be returned to the calling program. The initialize line control feature sets up instructions which check the line count before each line is printed and allows the user to skip to the top of the page and print a heading if the page is full.

Record Format

Input linkage from the user-register 14

POSITION	LENGTH	FIELD DESCRIPTION
0	1 byte	Length of message or lines per page (0 to skip to new page)
1 - 3	3 bytes	Address of message or user's page full routine
4	- -----	Normal return or TYPE retry return
8	- -----	TYPE ignore return

3. Generalized Sort Routine (XSORT)

Processing

XSORT is a small generalized in-core sort routine. Records are sorted in ascending sequential order and duplicate records are not eliminated. The macro XSRT is used to generate linkage to this routine. For example:

```
          XSRT  RCNT,BUFFER,8,4,2
+***      SORT  ROUTINE                       Sample Record
+         CNOP  0,4                           1 2 3 4 5 6 7 8
+         LH    0,RCNT
+         BAL   1,*+12                           Sort Field
+         DC    A(BUFFER)             Address of records to sort
+         DC    AL2(8)                Length of each term
+         DC    AL1(4)                Start of sort field
+         DC    AL1(2)                Length of sort field
+         L     15,=V(XSORT)
+         BALR  14,15
```

The maximum size record that can be sorted is 200 bytes. The routine base register (15) and the return address register (14) are both loaded by the user.

Record Format

Input linkage from the user consists of the count of records to sort in register 0 and register 1 points to the following:

POSITION	LENGTH	FIELD DESCRIPTION
0 - 3	4 bytes	Address of records to sort
4 - 5	2 bytes	Length of each record
6	1 byte	Start position of sort field
7	1 byte	Length of sort field

4. Dictionary Disk Look Up Routine (IKDICT)

Processing

During initialization, the communication vector is read from cylinder 1, track 0 of the dictionary disk file into core. If the condensed dictionary is not mounted, the note MOUNT FDA DICT DISK is typed on the console typewriter. The job will continue after the correct disk is mounted. When searching for a term, first the master index is scanned to find which cylinder index to scan. If this cylinder index is not already in core, it is read into core. The selected cylinder index table is scanned to find which cylinder to scan. If the track index for this cylinder is not in core, it is read into core. The selected track index is scanned to find which track to scan. If the selected track is not in core, it is read into core. As this track is scanned, if there is a track index or a record which is continued from the last track, the alternate return address is taken. Otherwise, the entire condensed dictionary record is moved to the output area. If this record is continued on the next track, the next track is read and the rest of the condensed dictionary record is put in the output area.

This disk look up routine is used by the MFEDIT, DUMPCELL, and SEARCH programs in this system.

Record Formats

The dictionary disk file consists of one physical record per disk track. Several of the dictionary data records are packed into one disk track. A track index is the first logical record in each cylinder except in cylinders 50, 100, and 150 where it is the first logical record on the third track. The cylinder indexes are on cylinders 50, 100, and 150, tracks 0 and 1. The master index is the second logical record in track 0 of cylinder 1 and is followed by the communication vector. Each track of data is defined as folflow:

a. General Format

POSITION	LENGTH	FIELD DESCRIPTION
0 - 1	2 bytes	Total length of data in this track
2 - 3	2 bytes	Binary zeros
4	1 byte	Code (X'00'-dictionary record follows, X'01'-dictionary record which follows is continued on next track, X'02'-dictionary record continued from last track follows, X'22'-master index follows, X'82'-cylinder index follows, X'43'-track index follows, X'FF'-Communication vector follows)
5 - 6	2 bytes	Length of this record
7-	(variable)	Dictionary record or index table depending upon code
-	1 byte	Code (same as byte 4 above - a code precedes and defines each record on the track)

b. Cylinder 1, track 0 Format

This track contains the first track index, the master index, and the communication vector.

POSITION	LENGTH	FIELD DESCRIPTION
0 - 1	2 bytes	Total record length
2 - 3	2 bytes	Binary zeros
4	1 byte	Code X'82'-track index
5 - 6	2 bytes	Length to next code
7 -658	652 bytes	Track index
659	1 byte	Code X'22'-Master index
660 -661	2 bytes	Length to next code
662 -1183	522 bytes	Master index
1184	1 byte	Code X'FF'-Communication vector
1185 -1186	2 bytes	Length of communication vector
1187 -1384	198 bytes	Communication vector

c. Index Format

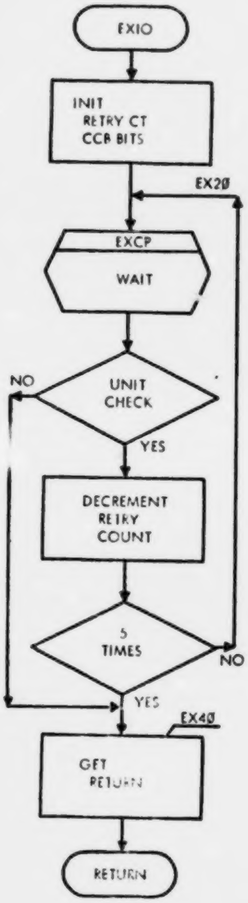
For master, cylinder, and track indexes.

POSITION	LENGTH	FIELD DESCRIPTION
0 - 1	2 bytes	Length of this index
2	1 byte	Length of first term
3-	(variable)	First term
-	1 byte	Length of second term
-	(variable)	Second term
	.	
	.	
	.	

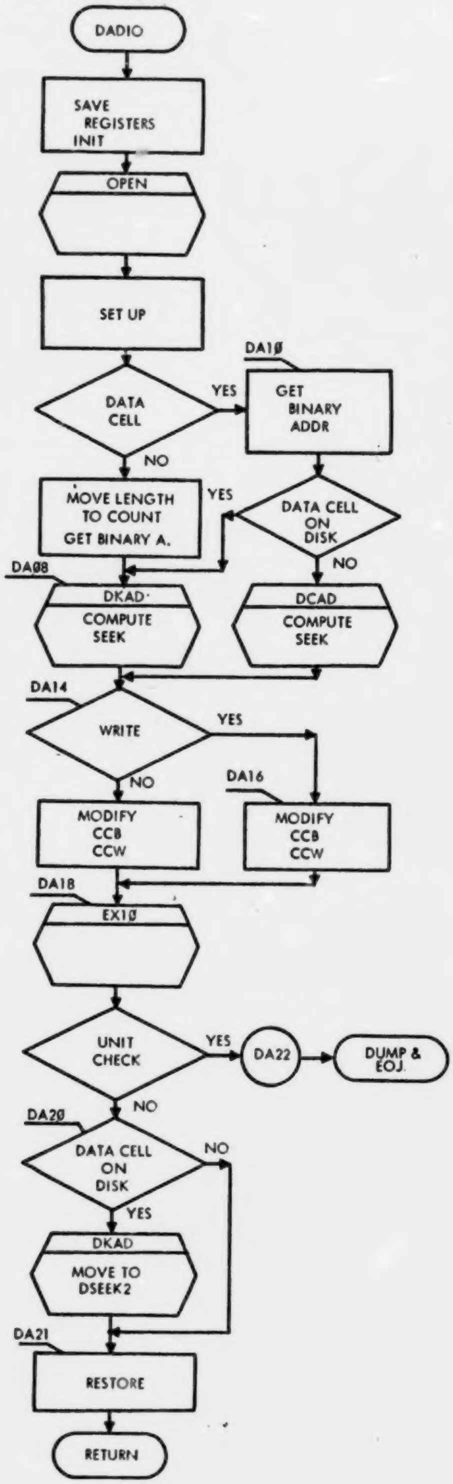
d. Communication Vector Format

POSITION	LENGTH	FIELD DESCRIPTION
0 - 7	8 bytes	Name of file (FDADICT)
8 - 15	8 bytes	Date file created (MM/DD/YY)
16 - 17	2 bytes	Track address of first cylinder index
18 - 19	2 bytes	Second track address of first cylinder index (unused)
20 - 21	2 bytes	Track address of second cylinder index
22 - 23	2 bytes	Second track address of second cylinder index (unused)
24 - 79	56 bytes	Track addresses for cylinder indexes 3,4,...
80 - 81	2 bytes	Last track written
82 - 87	6 bytes	Unused
88 - 89	2 bytes	Maximum length of disk track record
90 -197	108 bytes	Unused

EXECUTE CHANNEL PROGRAM

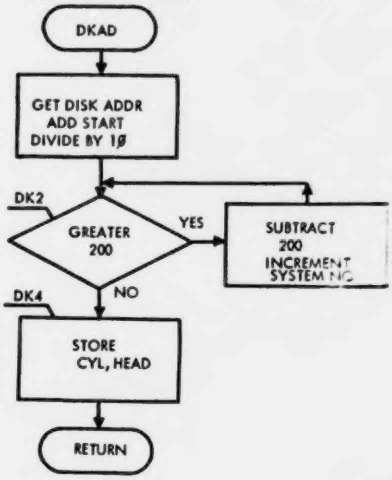


EXIO
INIT
RETRY CT
CCB BITS
EX20
EXCP
WAIT
UNIT
CHECK
NO
YES
DECREMENT
RETRY
COUNT
5
TIMES
NO
YES
EX40
GET
RETURN
RETURN



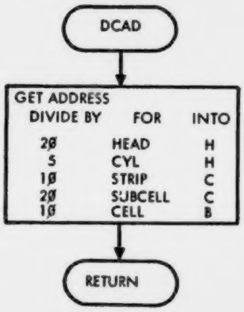
DADIO
SAVE
REGISTERS
INIT
OPEN
SET UP
DATA
CELL
YES
DA10
GET
BINARY
ADDR
NO
MOVE LENGTH
TO COUNT
GET BINARY A.
YES
DATA CELL
ON
DISK
NO
DA08
DKAD
COMPUTE
SEEK
DCAD
COMPUTE
SEEK
DA14
WRITE
YES
NO
DA16
MODIFY
CCB
CCW
MODIFY
CCB
CCW
DA18
EX10
UNIT
CHECK
YES
DA22
DUMP &
EOJ.
NO
DA20
DATA CELL
ON
DISK
NO
YES
DKAD
MOVE TO
DSEEK2
DA21
RESTORE
RETURN

COMPUTE DISK ADDRESS DADIO

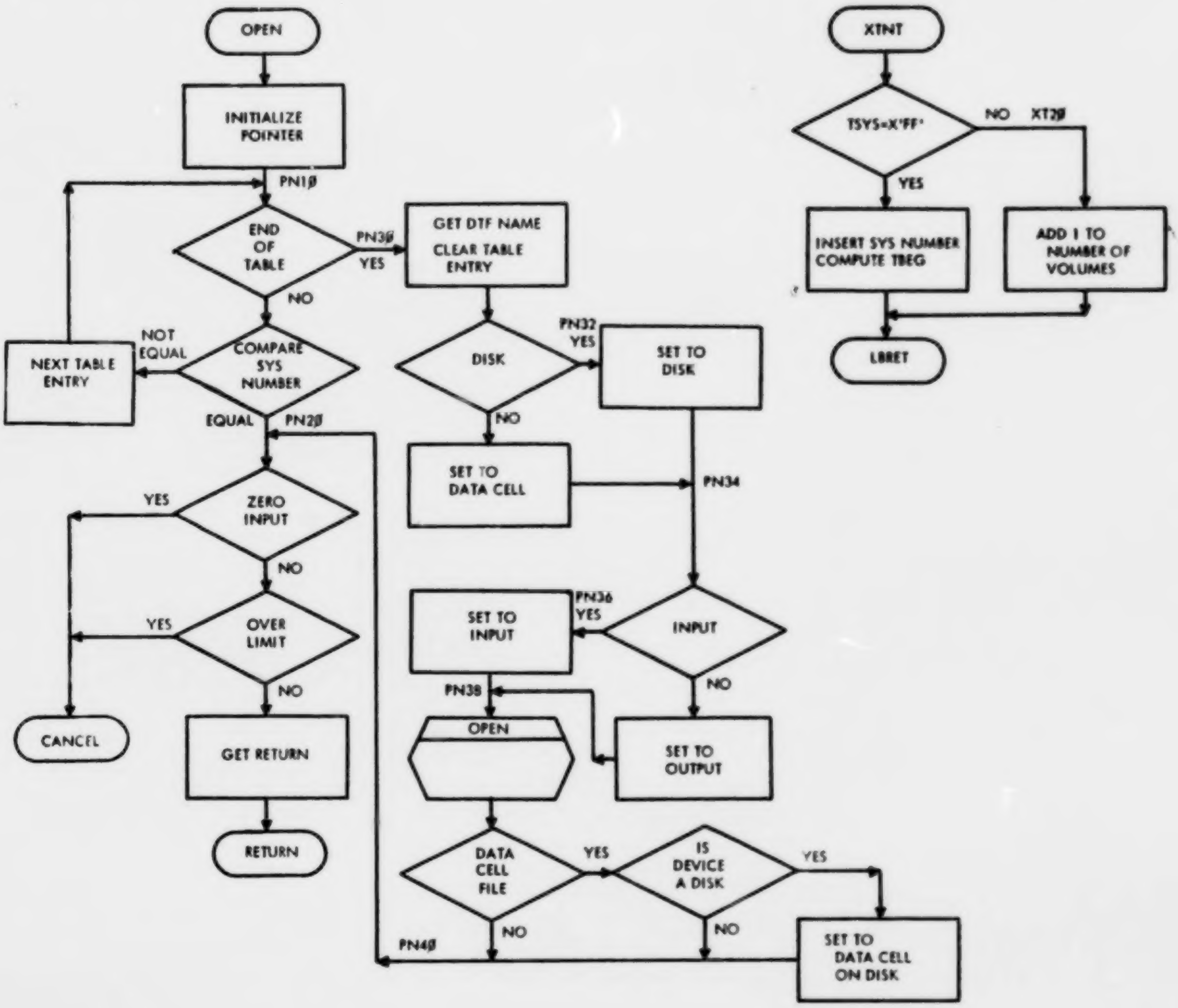


DKAD
GET DISK ADDR
ADD START
DIVIDE BY 10
DK2
GREATER
200
YES
SUBTRACT
200
INCREMENT
SYSTEM NO
NO
DK4
STORE
CYL, HEAD
RETURN

COMPUTE DATA CELL ADDRESS



DCAD
GET ADDRESS
DIVIDE BY FOR INTO
20 HEAD H
5 CYL H
10 STRIP C
20 SUBCELL C
10 CELL B
RETURN


OPEN
INITIALIZE POINTER
PN10
END OF TABLE
PN30
YES
NO
GET DTF NAME
CLEAR TABLE ENTRY
NEXT TABLE ENTRY
NOT EQUAL
COMPARE SYS NUMBER
EQUAL
PN20
DISK
PN32
YES
SET TO DISK
NO
SET TO DATA CELL
PN34
YES
ZERO INPUT
NO
YES
OVER LIMIT
NO
CANCEL
GET RETURN
RETURN
SET TO INPUT
PN36
YES
INPUT
NO
PN38
OPEN
SET TO OUTPUT
DATA CELL FILE
YES
IS DEVICE A DISK
YES
NO
NO
PN40
SET TO DATA CELL ON DISK
XTNT
TSYS=X'FF'
NO
XT20
YES
INSERT SYS NUMBER
COMPUTE TBEG
ADD 1 TO NUMBER OF VOLUMES
LBRET

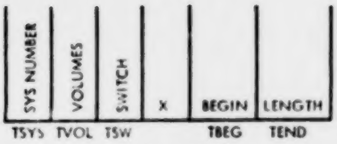
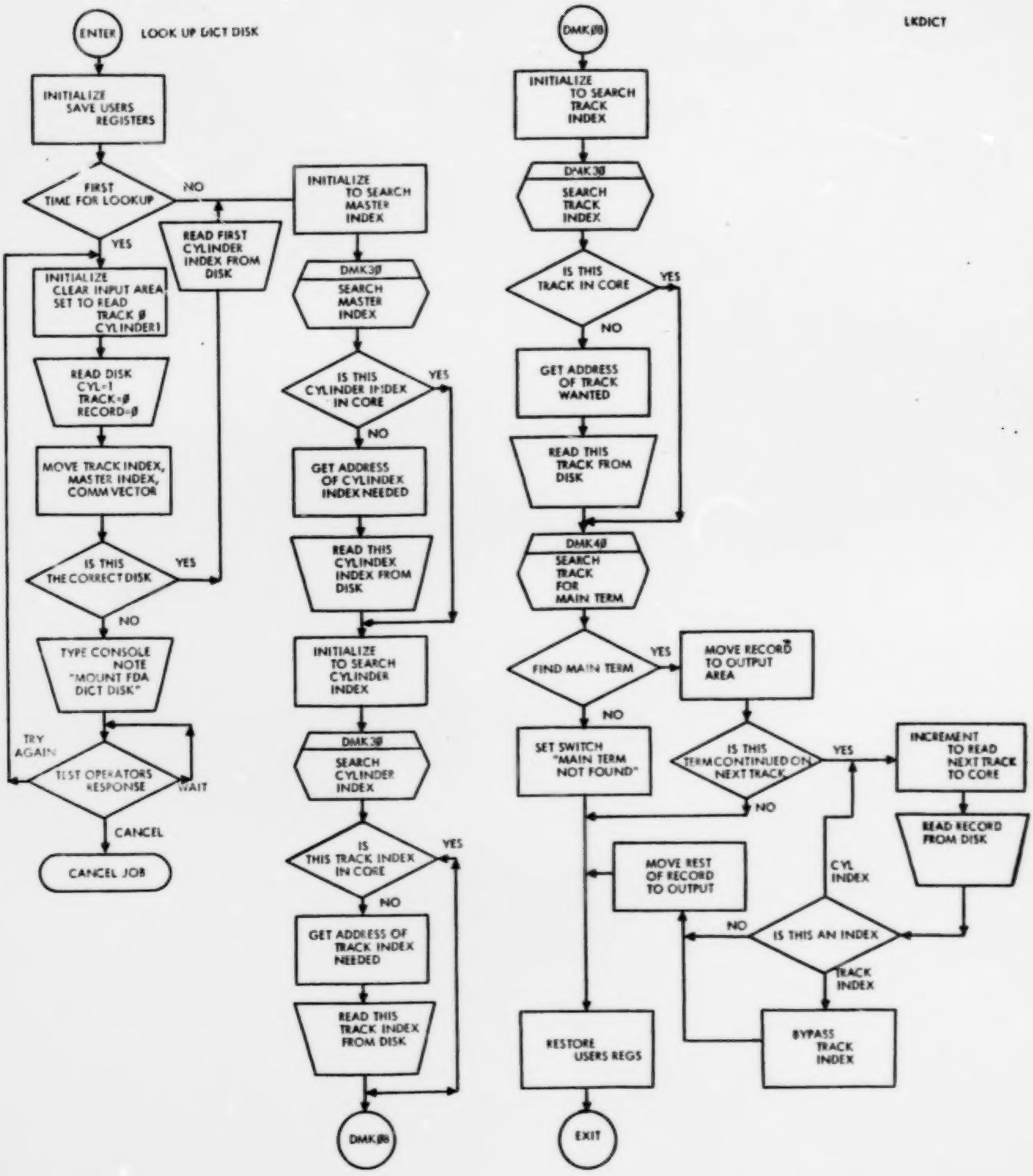
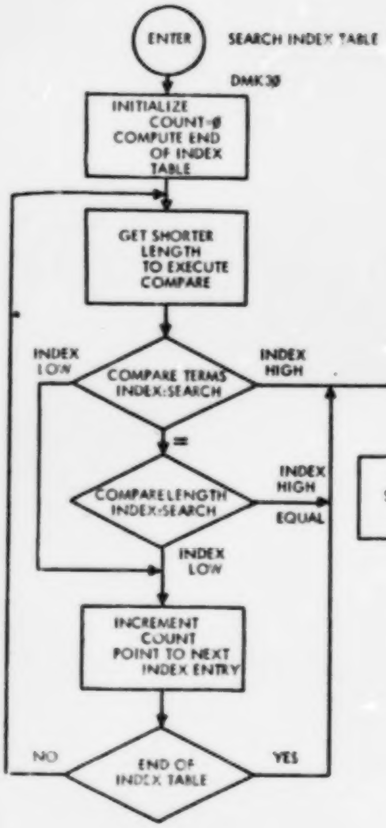

SYS NUMBER
VOLUMES
SWITCH
X
BEGIN
LENGTH
TSYS
TVOL
TSW
TBEG
TEND

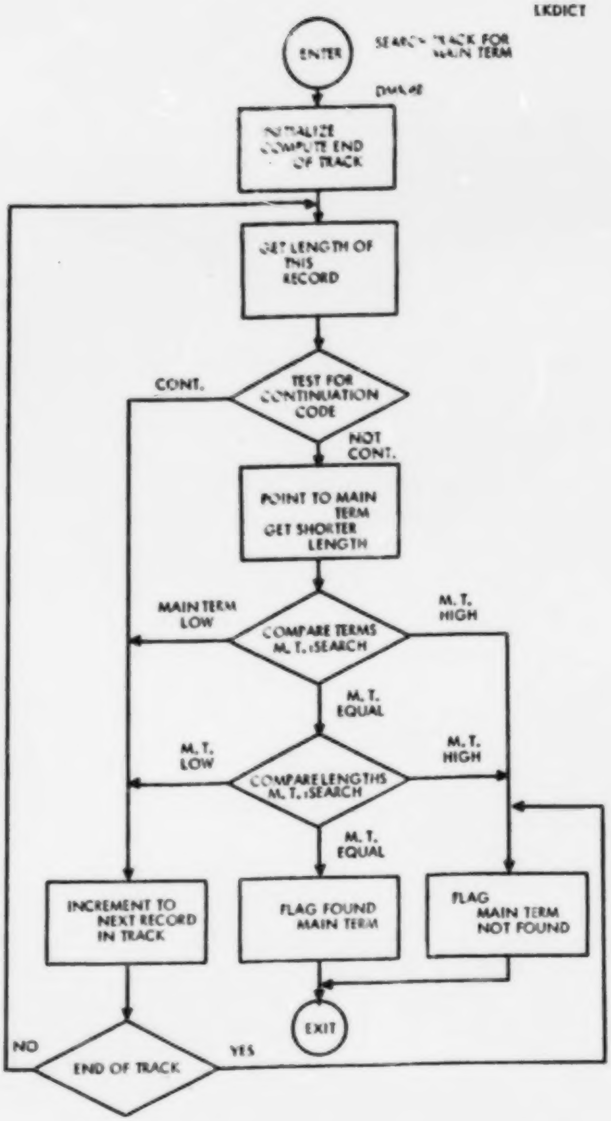
TABLE ENTRY


ENTER
LOOK UP DICT DISK
LKDICT
INITIALIZE SAVE USERS REGISTERS
FIRST TIME FOR LOOKUP
NO
YES
INITIALIZE CLEAR INPUT AREA SET TO READ TRACK 0 CYLINDER1
READ DISK CYL=1 TRACK=0 RECORD=0
MOVE TRACK INDEX, MASTER INDEX, COMM VECTOR
IS THIS THE CORRECT DISK
YES
NO
READ FIRST CYLINDER INDEX FROM DISK
TYPE CONSOLE NOTE "MOUNT FDA DICT DISK"
TRY AGAIN
TEST OPERATORS RESPONSE
WAIT
CANCEL
CANCEL JOB
INITIALIZE TO SEARCH MASTER INDEX
DMK30
SEARCH MASTER INDEX
IS THIS CYLINDER INDEX IN CORE
YES
NO
GET ADDRESS OF CYLINDEX INDEX NEEDED
READ THIS CYLINDEX INDEX FROM DISK
INITIALIZE TO SEARCH CYLINDER INDEX
DMK30
SEARCH CYLINDER INDEX
IS THIS TRACK INDEX IN CORE
YES
NO
GET ADDRESS OF TRACK INDEX NEEDED
READ THIS TRACK INDEX FROM DISK
DMK00
DMK00
INITIALIZE TO SEARCH TRACK INDEX
DMK30
SEARCH TRACK INDEX
IS THIS TRACK IN CORE
YES
NO
GET ADDRESS OF TRACK WANTED
READ THIS TRACK FROM DISK
DMK40
SEARCH TRACK FOR MAIN TERM
FIND MAIN TERM
YES
NO
MOVE RECORD TO OUTPUT AREA
SET SWITCH "MAIN TERM NOT FOUND"
IS THIS TERM CONTINUED ON NEXT TRACK
YES
NO
INCREMENT TO READ NEXT TRACK TO CORE
READ RECORD FROM DISK
CYL INDEX
IS THIS AN INDEX
NO
TRACK INDEX
MOVE REST OF RECORD TO OUTPUT
BYPASS TRACK INDEX
RESTORE USERS REGS
EXIT

LKDICT



ENTER
SEARCH INDEX TABLE
DMK30
INITIALIZE COUNT=0 COMPUTE END OF INDEX TABLE
GET SHORTER LENGTH TO EXECUTE COMPARE
INDEX LOW
COMPARE TERMS INDEX:SEARCH
INDEX HIGH
=
COMPARELENGTH INDEX:SEARCH
INDEX HIGH
EQUAL
INDEX LOW
SAVE COUNTER
EXIT
INCREMENT COUNT POINT TO NEXT INDEX ENTRY
NO
END OF INDEX TABLE
YES



ENTER
SEARCH TRACK FOR MAIN TERM
DMK40
INITIALIZE COMPUTE END OF TRACK
GET LENGTH OF THIS RECORD
CONT.
TEST FOR CONTINUATION CODE
NOT CONT.
POINT TO MAIN TERM GET SHORTER LENGTH
MAIN TERM LOW
COMPARE TERMS M. T.:SEARCH
M. T. HIGH
M. T. EQUAL
M. T. LOW
COMPARELENGTHS M. T.:SEARCH
M. T. HIGH
M. T. EQUAL
INCREMENT TO NEXT RECORD IN TRACK
FLAG FOUND MAIN TERM
FLAG MAIN TERM NOT FOUND
EXIT
NO
END OF TRACK
YES

TABLE BUILDING PROGRAM (TABBLD) PROCESSING LOGIC

The files are opened and the first card is read. The first card must be a valid program control card. If the first card is not a DCT*, FIL*, or CAT*, or if it is a CAT* with a non valid filename mnemonic an error message is printed and the job is cancelled. If the first card is a valid program control card a PHASE card is punched and the input cards for that table are read.

While the input cards are read and listed a running count is kept to ensure that no more than 256 cards are entered. As each card is read the number in columns 5-7 is checked. It must be between 000 and 255 and must not have been encountered before. If either of these conditions is violated an error switch is set. If the number is valid the mnemonic filename in columns 1-3 is checked for validity. It must be unique and must contain no special characters or embedded blanks. If either of these conditions is violated an error switch is set. After each card has been processed the error switch is interrogated and, if necessary, diagnostic messages are printed out. The program continues to read, list and validate the input data until an end-of-file condition is sensed on the card reader.

After all the input cards have been processed the program tests the error switch. If any errors have been encountered a message is printed and the job is cancelled. If there were no input data errors and if there were not more than 256 cards in the data set(which would also result in cancellation) the data is sorted on the mnemonic in columns 1-3.

After the sorting has been completed the program interrogates the counter. If less than 256 cards were included TABBLD pads the output table, since later programs require a full 256 entries. Then the output cards are punched which will be loaded and cataloged into the Core Image Library. All necessary cards are punched including the Job Control Language. The entire deck need only be placed in the card reader as a separate job. When the cards have all been punched, processing terminates.

Input

1. Requirements

   A control card must come first, followed by from 1 to 256 input cards.

2. Format

   a. Control Card

      For each run there must be one and only one control card. There are three permissible formats, all of which begin in column 1. They are:

      1. DCT*
      2. FIL*
      3. CAT*NNN — where NNN represents a file name. Please refer to the description of input cards for a more detailed specification.

   b. Input Cards

      Input cards have the following format:

      1. NNN XXX — where NNN represents the file name mnemonic. The first character which appears in column 1 must be alphabetic. The other characters may be alphabetic or numeric with trailing blanks. No special characters or embedded blanks are allowed.

         Column 4 is left blank.

         XXX which is punched in columns 5 through 7 symbolizes any number from 000 to 255. Each number must be used only once in each data set, but the numbers need not be in sequence.

         The remaining columns will normally be left blank, but they may be used for comments. The program does not check them in any way but they will appear on the printed listing of input.

Output

1. Card Formats and Explanatory Notes

   a. Card Formats

      1. // JOB CATALOG

      2. // OPTION CATAL,LINK

      3. PHASE DCTTAB,*
         PHASE FILTAB,*
         PHASE CATNNN,8

         The characters in columns 10 through 12 correspond to columns 1 through 3 of the input control card. NNN represents the filename mnemonic characters which appear in columns 5 through 7 of a CAT* type input card.

      4. // EXEC ASSEMBLY

      5. CSECT

      6. DC C'XXX'
         DC FL1'XXX'

         The XXX's in this pair of cards represent the filename mnemonic and corresponding number. The number of pairs of DC cards will be equal to the number of input cards.

      7. DC NNNX'FFFFFFFF'

         A filler card is punched only if there were fewer than 256 input cards. NNN equals the difference between 256 and the number of input cards.

      8. /*

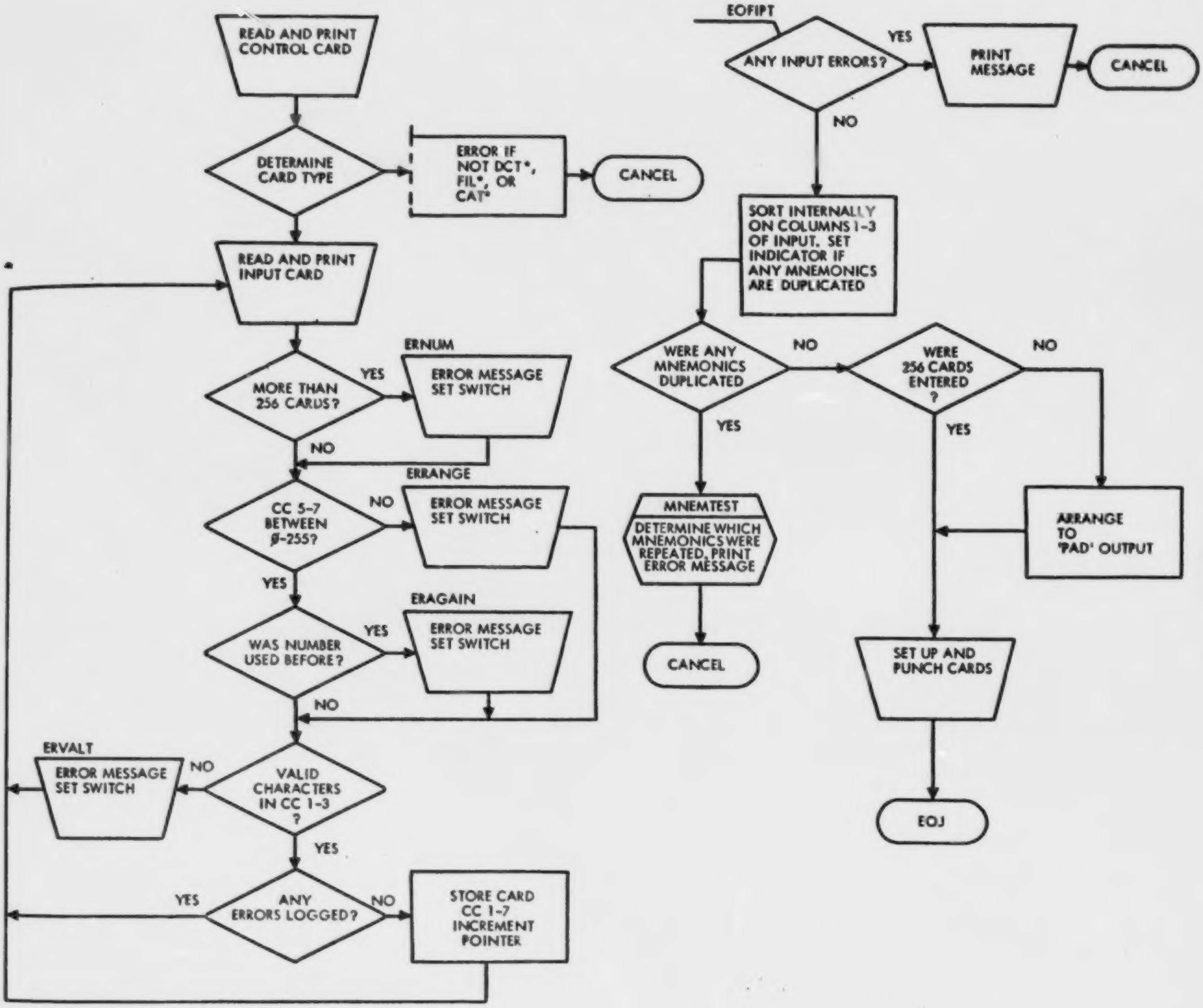
      9. // EXEC LNKEDT

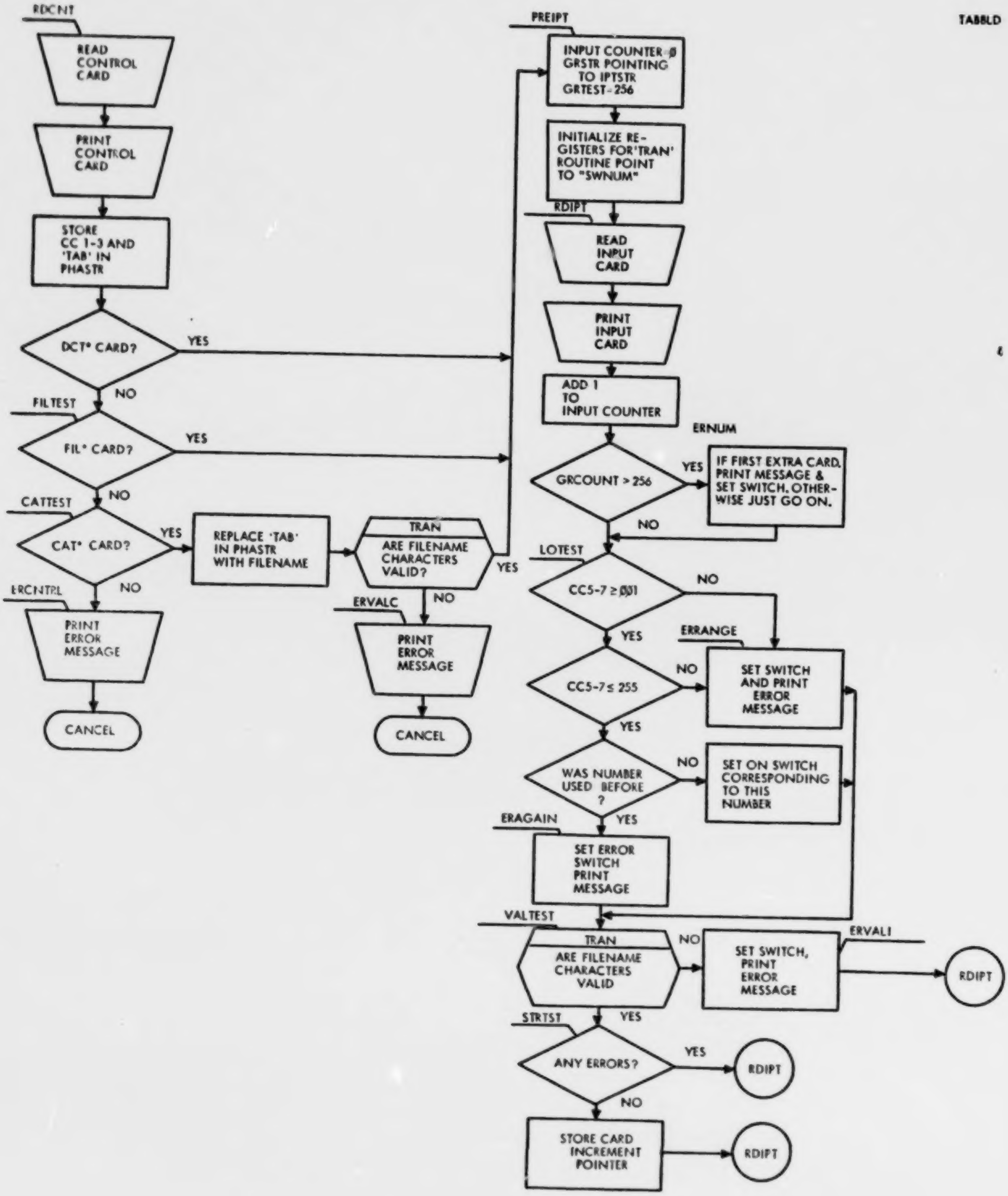
      10. /&

2. Listing of Sample Output

   For the protection of the user, this program produces punched output only when all of the input was valid. A listing of the input to the program is always produced on SYSLST. If any errors have been encountered, messages indicating the nature of the errors are also printed.

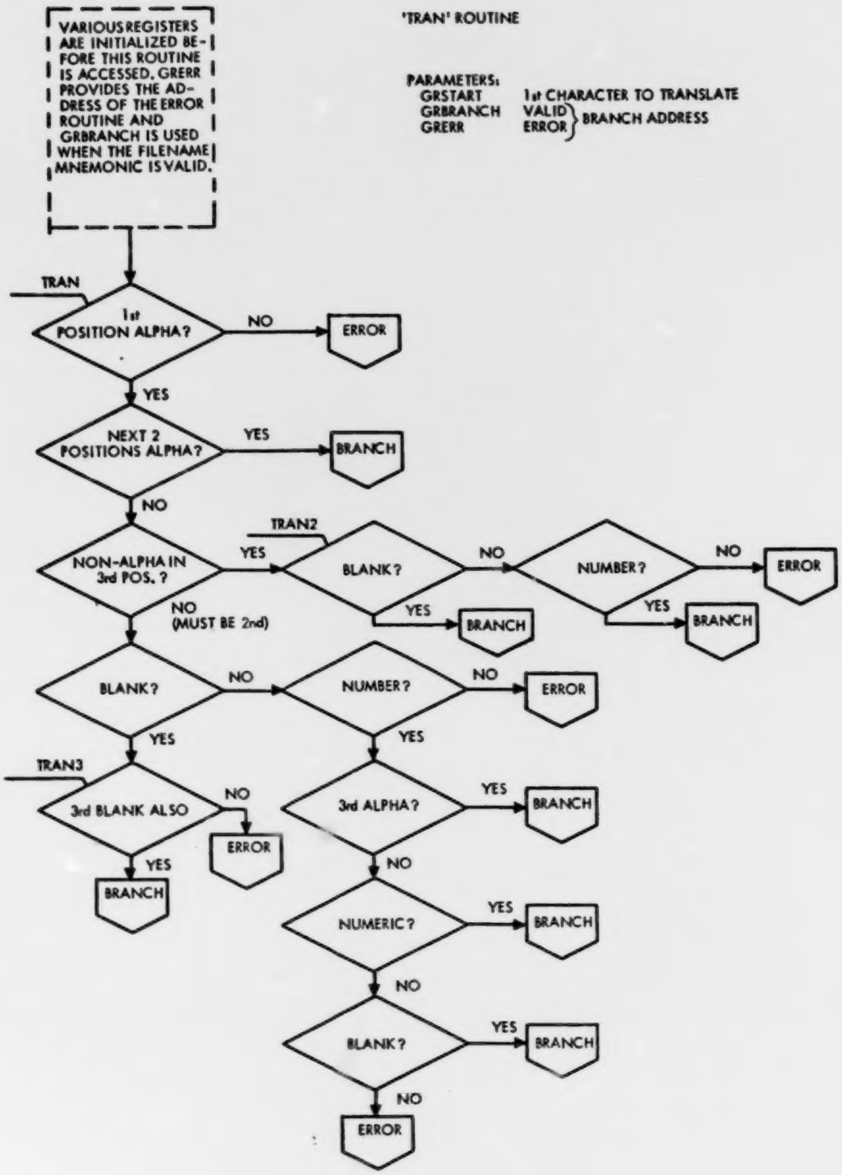
GENERAL FLOW CHART

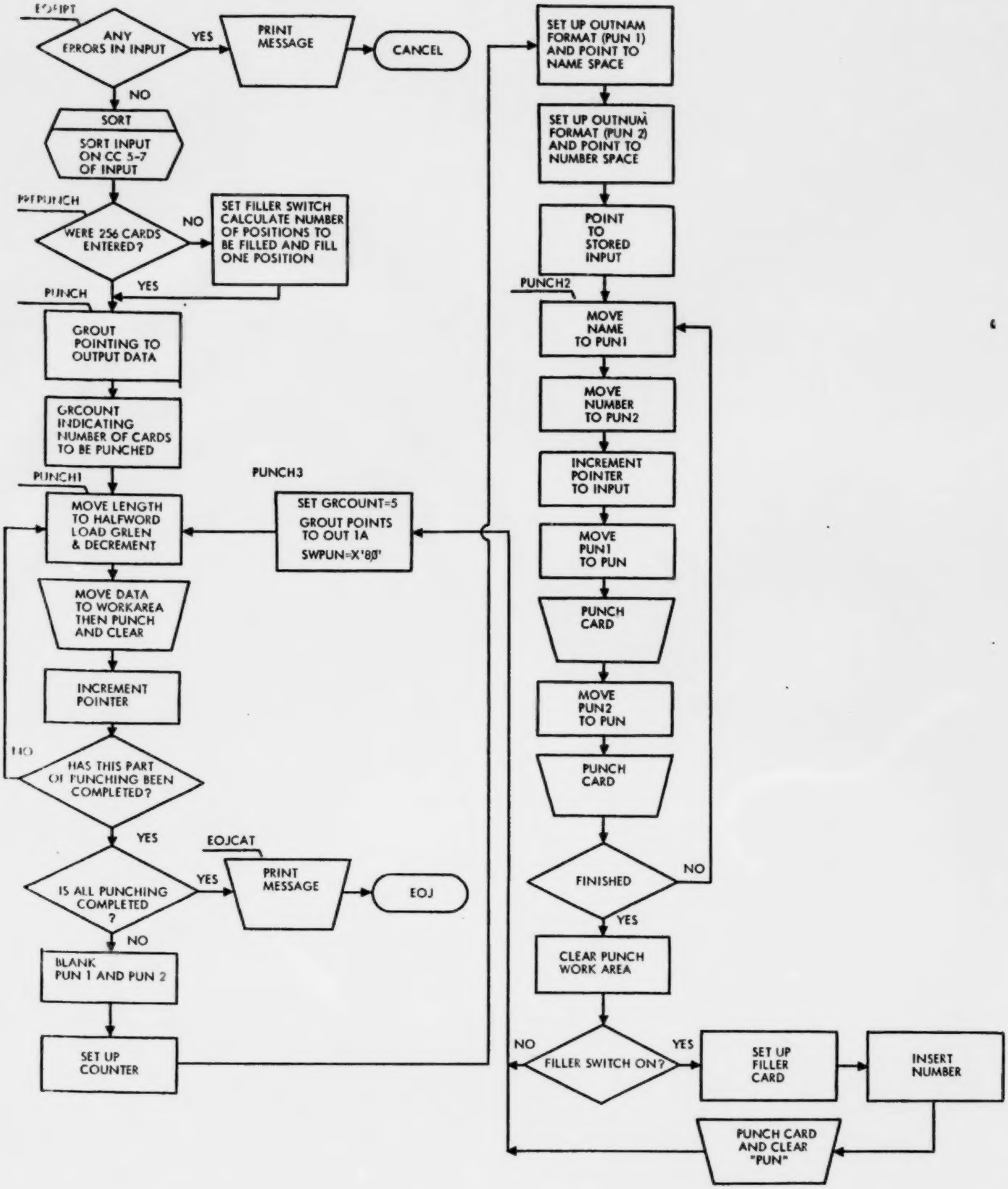

READ AND PRINT CONTROL CARD
DETERMINE CARD TYPE
ERROR IF NOT DCT*, FIL*, OR CAT*
CANCEL
READ AND PRINT INPUT CARD
ERNUM
MORE THAN 256 CARDS?
YES
ERROR MESSAGE SET SWITCH
NO
ERRANGE
CC 5-7 BETWEEN 0-255?
NO
ERROR MESSAGE SET SWITCH
YES
ERAGAIN
WAS NUMBER USED BEFORE?
YES
ERROR MESSAGE SET SWITCH
NO
ERVALT
ERROR MESSAGE SET SWITCH
NO
VALID CHARACTERS IN CC 1-3 ?
YES
YES
ANY ERRORS LOGGED?
NO
STORE CARD CC 1-7 INCREMENT POINTER
EOFIPT
ANY INPUT ERRORS?
YES
PRINT MESSAGE
CANCEL
NO
SORT INTERNALLY ON COLUMNS 1-3 OF INPUT. SET INDICATOR IF ANY MNEMONICS ARE DUPLICATED
WERE ANY MNEMONICS DUPLICATED
NO
WERE 256 CARDS ENTERED ?
NO
YES
YES
MNEMTEST
DETERMINE WHICH MNEMONICS WERE REPEATED, PRINT ERROR MESSAGE
CANCEL
ARRANGE TO 'PAD' OUTPUT
SET UP AND PUNCH CARDS
EOJ

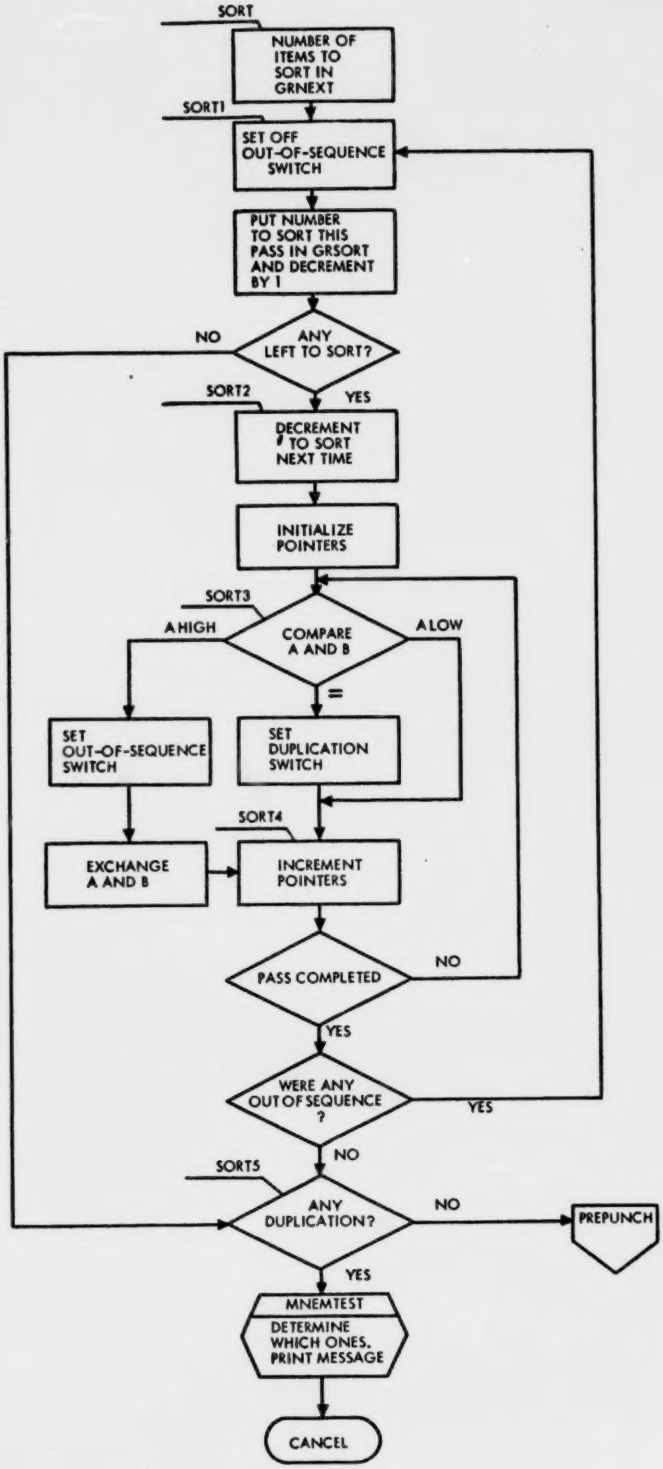

TABBLD
RDCNT
READ CONTROL CARD
PRINT CONTROL CARD
STORE CC 1-3 AND 'TAB' IN PHASTR
DCT* CARD?
YES
NO
FILTEST
FIL* CARD?
YES
NO
CATTEST
CAT* CARD?
YES
NO
REPLACE 'TAB' IN PHASTR WITH FILENAME
TRAN
ARE FILENAME CHARACTERS VALID?
YES
NO
ERCNTRL
PRINT ERROR MESSAGE
CANCEL
ERVALC
PRINT ERROR MESSAGE
CANCEL
PREIPT
INPUT COUNTER=0 GRSTR POINTING TO IPTSTR GRTEST=256
INITIALIZE REGISTERS FOR 'TRAN' ROUTINE POINT TO "SWNUM"
RDIPT
READ INPUT CARD
PRINT INPUT CARD
ADD 1 TO INPUT COUNTER
ERNUM
GRCOUNT > 256
YES
IF FIRST EXTRA CARD, PRINT MESSAGE & SET SWITCH, OTHERWISE JUST GO ON.
NO
LOTEST
CC5-7 ≥ 001
NO
YES
ERRANGE
CC5-7 ≤ 255
NO
SET SWITCH AND PRINT ERROR MESSAGE
YES
WAS NUMBER USED BEFORE ?
NO
SET ON SWITCH CORRESPONDING TO THIS NUMBER
YES
ERAGAIN
SET ERROR SWITCH PRINT MESSAGE
VALTEST
TRAN
ARE FILENAME CHARACTERS VALID
NO
SET SWITCH, PRINT ERROR MESSAGE
ERVALI
RDIPT
YES
STRTST
ANY ERRORS?
YES
RDIPT
NO
STORE CARD INCREMENT POINTER
RDIPT

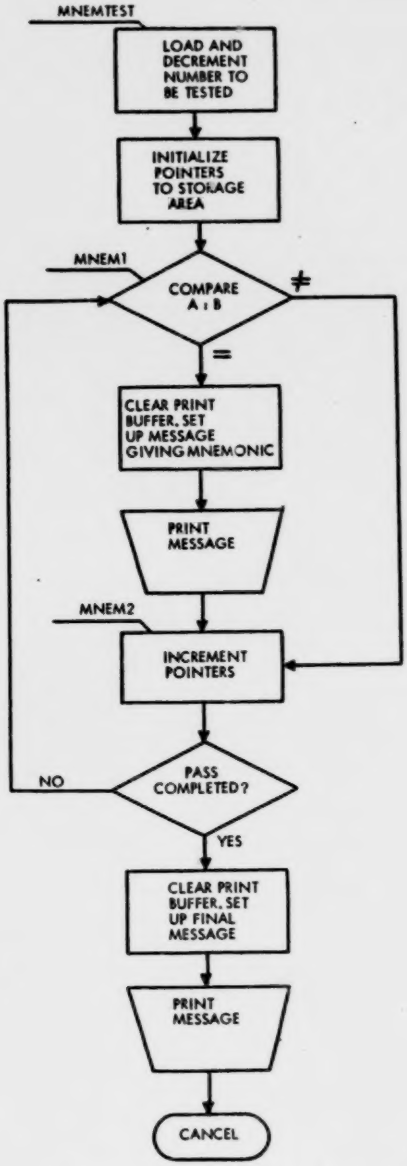
'TRAN' ROUTINE

PARAMETERS:
GRSTART 1st CHARACTER TO TRANSLATE
GRBRANCH VALID } BRANCH ADDRESS
GRERR ERROR }


VARIOUS REGISTERS ARE INITIALIZED BEFORE THIS ROUTINE IS ACCESSED. GRERR PROVIDES THE ADDRESS OF THE ERROR ROUTINE AND GRBRANCH IS USED WHEN THE FILENAME MNEMONIC IS VALID.
TRAN
1st POSITION ALPHA?
NO
ERROR
YES
NEXT 2 POSITIONS ALPHA?
YES
BRANCH
NO
TRAN2
NON-ALPHA IN 3rd POS. ?
YES
BLANK?
NO
NUMBER?
NO
ERROR
YES
BRANCH
YES
BRANCH
NO (MUST BE 2nd)
BLANK?
NO
NUMBER?
NO
ERROR
YES
YES
TRAN3
3rd BLANK ALSO
NO
ERROR
YES
BRANCH
3rd ALPHA?
YES
BRANCH
NO
NUMERIC?
YES
BRANCH
NO
BLANK?
YES
BRANCH
NO
ERROR


EOFIPT
ANY ERRORS IN INPUT
YES
PRINT MESSAGE
CANCEL
NO
SORT
SORT INPUT ON CC 5-7 OF INPUT
PREPUNCH
WERE 256 CARDS ENTERED?
NO
SET FILLER SWITCH CALCULATE NUMBER OF POSITIONS TO BE FILLED AND FILL ONE POSITION
YES
PUNCH
GROUT POINTING TO OUTPUT DATA
GRCOUNT INDICATING NUMBER OF CARDS TO BE PUNCHED
PUNCH1
MOVE LENGTH TO HALFWORD LOAD GRLEN & DECREMENT
PUNCH3
SET GRCOUNT=5
GROUT POINTS TO OUT 1A
SWPUN=X'80'
MOVE DATA TO WORKAREA THEN PUNCH AND CLEAR
INCREMENT POINTER
NO
HAS THIS PART OF PUNCHING BEEN COMPLETED?
YES
EOJCAT
IS ALL PUNCHING COMPLETED ?
YES
PRINT MESSAGE
EOJ
NO
BLANK PUN 1 AND PUN 2
SET UP COUNTER
SET UP OUTNAM FORMAT (PUN 1) AND POINT TO NAME SPACE
SET UP OUTNUM FORMAT (PUN 2) AND POINT TO NUMBER SPACE
POINT TO STORED INPUT
PUNCH2
MOVE NAME TO PUN1
MOVE NUMBER TO PUN2
INCREMENT POINTER TO INPUT
MOVE PUN1 TO PUN
PUNCH CARD
MOVE PUN2 TO PUN
PUNCH CARD
FINISHED
NO
YES
CLEAR PUNCH WORK AREA
NO
FILLER SWITCH ON?
YES
SET UP FILLER CARD
INSERT NUMBER
PUNCH CARD AND CLEAR "PUN"


SORT
NUMBER OF ITEMS TO SORT IN GRNEXT
SORT1
SET OFF OUT-OF-SEQUENCE SWITCH
PUT NUMBER TO SORT THIS PASS IN GRSORT AND DECREMENT BY 1
NO
ANY LEFT TO SORT?
YES
SORT2
DECREMENT TO SORT NEXT TIME
INITIALIZE POINTERS
SORT3
A HIGH
COMPARE A AND B
A LOW
=
SET OUT-OF-SEQUENCE SWITCH
SET DUPLICATION SWITCH
SORT4
EXCHANGE A AND B
INCREMENT POINTERS
PASS COMPLETED
NO
YES
WERE ANY OUT OF SEQUENCE ?
YES
NO
SORT5
ANY DUPLICATION?
NO
PREPUNCH
YES
MNEMTEST
DETERMINE WHICH ONES. PRINT MESSAGE
CANCEL


MNEMTEST
LOAD AND DECREMENT NUMBER TO BE TESTED
INITIALIZE POINTERS TO STORAGE AREA
MNEM1
COMPARE A : B
≠
=
CLEAR PRINT BUFFER, SET UP MESSAGE GIVING MNEMONIC
PRINT MESSAGE
MNEM2
INCREMENT POINTERS
NO
PASS COMPLETED?
YES
CLEAR PRINT BUFFER, SET UP FINAL MESSAGE
PRINT MESSAGE
CANCEL

DICTIONARY EDIT (DEDIT) PROCESSING LOGIC

The computer operator is asked, via a typed message, for the input device. CARD and TAPE are the only acceptable replies and will be repeatedly requested until one is given. The proper input file, a printer file and a tape output file are then opened. The first input record is read. If this is not a BCH or HDR record, an error message is typed, and the job is terminated.

A BCH record indicates that block change input records will follow. There are two acceptable input record types for block change data - DELETE and REPLACE. Both of these types may have continuation records.

As each record is read in it is sequence checked. This input record is printed.

If the record read is not a continuation record, a check is made of the process switch to see if any information has been compressed. If there is such information, it is now edited. Such items as term length, use, and punctuation are checked at this time. If anything is found to be incorrect, it will be noted by one of the asterisked error messages on the following page.

Block change errors prevent any tape output - both of block change and dictionary maintenance information.

If there were no errors, the reformatted information will be put out on tape. Each DELETE or REPLACE data set will generate only one output record.

If the input record was a DELETE or REPLACE record, the compression area is cleared to X'00', the process switch is set and the information on the input record is compressed (i.e. all extraneous blanks are removed). If the record was a continuation record, it, too, is compressed, the first character being placed immediately after the last character of the previous record. If the record is a BCT, the next input record is read. It must be either a HDR or an EOF. If the input record was none of those mentioned, it is in error and diagnosed as such.

Once a HDR record is read, only dictionary maintenance input is acceptable.

As each record is read, it is sequence checked. Next, the record type is determined. It may be a HDR, ADD, DELETE, REPLACE, SCOPE or continuation record. What happens in each of these situations is outlined below.

```
  NOTE 001 INFORMATION IN CC. 6-10 OF CONTINUATION CARD.
  NOTE 006 A BLOCK CHANGE DATA SET MAY NOT BEGIN WITH A CONTINUATION CARD.
  NOTE 007 BLOCK CHANGE CARD OUT OF SEQUENCE.
  NOTE 008 INVALID BLOCK CHANGE CARD.  COLUMN 5 IS NOT BLANK, YET CARD IS NOT DELETE, REPLACE OR BCT.
* NOTE 031 BLOCK CHANGE CARD OPTION CONTAINS AN INVALID CHARACTER.
* NOTE 032 INVALID OPTION ON BLOCK CHANGE CARD (LESS THAN 2 OR GREATER THAN 3 CHARACTERS IN LENGTH).
* NOTE 040 THE ONLY VALID 2-CHARACTER, BLOCK CHANGE OPTIONS ARE SA OR RT.
* NOTE 065 TERM NAME ON BLOCK CHANGE CARD IS MORE THAN 64 CHARACTERS LONG.
* NOTE 066 TERM NAME ON BLOCK CHANGE CARD HAS NO TERMINATING COLON.
* NOTE 067 BLOCK CHANGE TERM NAME CONTAINS AN INVALID CHARACTER.
```

1. HDR

   a. Test process switch to see if SCOPE was last record processed. If it was, go to D.

   b. Has the process switch been set? If no, go to D.

   c. Go to DCTBLD. Here the process switch is tested to see whether the compressed data set is an ADD,DELETE or REPLACE. All information is then edited in SCAN. When all data has been edited, the numbers of any errors found in this data set are printed in NOTENUM. If there were no serious errors in this data set, and if there had been no block change errors, 119-character partial records are formed in FORMREC.

   d. Print the message for errors found in all data sets for the term just processed. If there were no serious errors found for this term and if there had been no block change errors, the 119-character partial records previously formed are expanded to full 185 character output records and are written on tape. All this is done in MESSPRT.

      Re-initialize switches for processing the new term.

   e. Move input record to print area; print this record.

   f. Is the rest of the HDR record blank? If yes, go to H.

   g. Take term name off the HDR, eliminating all unnecessary blanks. If the name is more than 64 characters long, log an error. Otherwise, move the name and its length byte into the term name holder. Set the old term switch and general switch to indicate a name found.

   h. Read next record.

2. ADD, DELETE or REPLACE

   a. Was last record processed a SCOPE? If no, go to C.

   b. Log error(SCOPE may only be followed by SCOPE or HDR.) Go to F.

   c. Is the process switch set? If no, go to E.

   d. Go to DCTBLD. See 1 - C for details.

e. Set process switch for this data set: ADD=X'01', DELETE=X'02'; REPLACE=X'13'. Clear compression area to X'00'. Compress input record - i.e. remove all extraneous blanks.

f. Move input record to print area and print.

g. Read next record.

3. SCOPE

a. Was the last record a SCOPE record? If yes, go to C.

b. Go to DCTBLD. See I-C for details.

c. Set process switch to X'08' for SCOPE. Check the general switch to see if a term name has been found. If there isn't one, log an error.

d. Is the separation column (13) blank? If no, log an error and go to J.

e. Is the NNN between 001 and 255? If no, log an error and go to J.

f. Scope notes are not compressed. Are there any non-blank characters after col. 63? If so, log an error and go to J.

g. Have there been any block change errros or errors in this data set? If so, go to J.

h. Form a 119-character partial output record and move it to the core hold area.

j. Move input record to print area and print.

k. Read next record.

4. Continuation

a. Is the process switch set? If no, go to E.

b. Was the last record processed a SCOPE record? If yes, log an error and go to E.

c. Are columns 6-10 blank? If no, log an error.

d. Compress the information in columns 11-80. In the compression area, the first character in this record will be placed immediately after the last character of the previous record.

e. Move the input record to the print area and print.

f. Read the next record.

FILECK

This routine is used to validate dictionary, file and context entries. It takes the DCT, FIL or CXT input entry, adds trailing blanks if three characters were not specified, and checks this three character code against the proper table. There is one dictionary table, one file table, and as many context tables as there are defined files in the file table. The dictionary and file tables are always held in core. Only one context table is held in core. Whenever a CXT is found, a check is made to determine if the proper table is in core. If not, the required table is loaded.

A binary search is used to match the input entry with a table entry. If a match is made, the binary number which corresponds to the entry is saved for placement in the output record. If no match is found, an error bit is set.

FORMEC

This routine checks in the option switch for those data types which have been found during the editing process.

X'FF' -Indicates a DELETE - ALL record is to be formed.
X'01' -Function record is to be formed.
X'04' -One or more related terms have been found. These have been stored in a hold area in core. One output record is made for each term.* If there were any FC entries associated with this term, they, too, are placed in the output record.
X'08' -One or more see also terms have been found. These are handled in the same manner as the related terms.*
X'10' -A USE term was found; form a use record.*
X'20' -One or more dictionaries have been indicated for this term. One record is made indicating the 1-20 dictionaries specified.
X'80' -There have been File/Context specifications for the main term. Their binary equivalents are stores in a core hold area. One record is made indicating the 1-20 FC specified.

As the data types are encountered, 119-character partial records are build and placed in a core hold area. If more than 60 partial records are built, an error is logged and output record building is eliminated for this data set.

*In REPLACE data sets, term names occur in pairs. The first is the name to be replaced and generates a delete record. The second is the replacing name and generates an add record.

MESSPRT

Whenever a HDR record, first SCOPENNN record of a series or an EOF is encountered a check of the error message bytes is made. Each bit in the fifteen bytes corresponds to an error note. Any time a bit is found to be on (ie.a 1) the corresponding error message is printed. If any bit of the first twelve bytes is found to be on, no records will be put out for the term just processed. A switch (NOREC) is set to indicate this. Bits of the last three error message bytes correspond to warning error messages. These do not prevent output records from being made. Before the routine is exited, the error message bytes are set to all X'00'.

NOTENUM

This routine shares logic with MESSPRT. Its function is to indicate to the user the data set in which an error occurred.

Whenever a non-continuation card is encountered (other than the 2-n scope notes in a series) a check of the error number bytes is made. If a bit is found to be on, the number of the corresponding message is placed in the print buffer. These numbers are printed 25 at a time.

When the routine is entered, the error number bytes are 'OR'ed' with the error message bytes. When the routine is exited, the error number bytes are set to all X'00'.

Once all the message error bytes have been checked, NOREC is tested. If it is not set, tape records are now formed and written. Each of the 119 character partial records formed in FORMREC is moved to the tape output work area immediately after the record byte (X'01'), length byte for the main term, and the main term name padded with blanks to 64 positions. The completed 185 character output record is then written. A X'FF' in the partial record hold area indicates that all records have been formed.

SCAN

This routine edits the dictionary maintenance input which has had all extraneous blanks removed. This information is checked against a translate table. Key characters in this process are the colon, semi-colon, and a X'00' which indicates the end of the information to be checked.

When a colon is found, the information immediately preceding it is assumed to indicate a data type, or option. Therefore, if this information is not F, FC, NT, RT, SA, USE, DICT, or SCOPE, an error is noted and a scan is made for the next colon. Additionally, certain combinations of these options found are invalid in DELETE and/or REPLACE data sets. For example, NT: will be diagnosed as an error any time it appears as other than the first option of an ADD data set.

Once the option has been identified, the scan continues. For NT:, RT:, SA: and USE:, the next key character is a semi-colon. If this is found, the length of the term name is determined and stored with the term name.

If the length is greater than 64, an error is logged. If a semicolon is not the next special character, an error is logged and a scan is made for the next colon.

For F: and DICT:, the next key character is a (. If this is found, the entries within the parentheses are checked for validity. Valid entries are indicated in the section on INPUT. These entries must be separated by semicolons. The options F: and DICT: are terminated by the combination);. If either of these characters is missing, or if an invalid character is found, an error is logged and a scan is made for the next colon.

For FC:, a dash(-) is the next key character. This separates the FIL from the CXT. The file and context entries are validated in FILECK. There must be a semicolon after the CXT entry.

SCOPE: Must be followed by ALL and X'00", which indicates the end of the information.

No error will be logged if a X'00' is found when a semicolon is expected, with the exception of semicolons within parentheses. This is also true after an error has been logged and a scan is being made for the next colon. A X'00' will always terminate the editing process.

Input

Input to this program is punched into cards according to the formats described below. General constraints which apply to DEDIT input are:

1. A dictionary term may consist of not more than 64 alpha-numeric characters. Seven special characters are also permissible: right and left parentheses, hyphen, slash, comma, period, and pound sign.
2. All input cards must be in ascending sequence based upon digits punched in columns 1 through 4. These numbers need not be consecutive.
3. All block change cards must precede any changes to individual dictionary records. Block change cards may be numbered independently from individual change records (i.e., block change cards may be numbered from 1 to n, immediately followed by individual term changes numbered from 1 to n).

4. The first input record must be a BCH (block change header) record or a HDR (dictionary term header) record.

5. All ADD, DELETE, and REPLACE records may be continuation cards. These continuation cards contain a sequence number in columns 1 through 4, with columns 5 through 10 blank. The continuation data begins in column 11.

6. If there is a character punched in column 80 of a card and a character punched in column 11 of a continuation card, it is assumed that there is no blank between the two characters.

7. All SCOPEnn(scope note) cards must be the last cards pertaining to a given dictionary term. Only another SCOPE card, relating to the same term, a HDR card -- indicating the beginning of a new dictionary term--or an EOF (end-of-file) card may follow a SCOPE card.

Block Change Cards

1. Block change header must be the first of the block change cards; only one is accepted. The format is:

Columns 1 - 4	Card Sequence Number
Columns 5 - 7	'BCH'
Columns 8 -80	Blank

2. Block change trailer must be the last of the block change cards; only one is accepted. The format is:

Columns 1 - 4	Card Sequence Number
Columns 5 - 7	'BCT'
Columns 8 -80	Blank

3. Delete

Columns 1 - 4	Card Sequence Number
Columns 5 -10	'DELETE'
Column 11	Blank
Columns 12 -80	Term name followed by a colon followed by 'ALL' or any combination of 'SA', 'RT', 'USE' separated by semicolons.

4. Replace

Columns 1 - 4	Card Sequence Number
Columns 5 -11	'REPLACE'
Column 12	Blank
Columns 13 -80	Term name to be replaced followed by a colon followed by the replacing name.

An example of two different block changes is given below:

```
0001BCH
0002DELETE ACUTE POLIOMYELITIS: RT; SA
0003REPLACE ACUTE POLIOMYELITIS: POLIOMYELITIS
0004BCT
```

These cards will result in the deletion of the term ACUTE POLIOMYELITIS wherever it appears in the dictionary file as either a related term (RT) or as a see also (SA) entry. If the term ACUTE POLIOMYELITIS appears other than as a RT or a SA entry in any dictionary record, it will be replaced by the single word POLIOMYELITIS, as directed by card 0003 above. Since the delete card above eliminated the term from all RT and SA occurrences, the only possible instance of replacement would be in the USE entries.

Note that the block change delete does not result in the elimination of the term as a main dictionary entry. The term is deleted only where it is embedded within the data under a main entry. To delete a main dictionary term it is necessary to use the dictionary maintenance cards described below. A main term cannot be deleted from the dictionary by use of the block change.

Dictionary Maintenance Cards

1. HDR Card

Columns 1 - 4	Card Sequence Number
Columns 5 - 7	'HDR' (designates the beginning of a set of data pertinent to a term)
Column 8	Blank
Columns 9 - 80	Blank if a new index term is being entered into the dictionary. Term name if maintaining existing dictionary term.

2. ADD Card

Columns 1 - 4	Card Sequence Number
Columns 5 - 7	'ADD' (indicates that a new term is being added to the dictionary or that information is being added to an existing term)
Column 8	Blank
Columns 9 - 80	As described below:

New Term	'NT: Term Name;'
Designation	Must be present if there was no name on the 'HDR' card. Must be the first of the 'ADD' card entries. Only one NT: is allowed per 'ADD' data set.

Function	PD 'F (TD;SD;ID);' CD If the term may be used as a descriptor in the master file, either PD,CD, or TD (never more than one) should be entered. A function must be given whenever NT: is present. Only one F: ( ) entry is allowed per data set. ***PD - Precise Descriptor ***TD - Trial Descriptor ***CD - Common Descriptor ***SD - Subdescriptor ***ID - Identification File Entry
Dictionary	"DICT: (DCT:DCT:...);' This entry specifies the dictionaries to which the term belongs. Up to 20 dictionaries may be specified for a term. Only one Dict: entry per 'ADD' data set. Each DCT is a 1-3 character entry.
File/Context	'FC: FIL-CXT;' This entry specifies the files in which a term may appear and the context in which it is used within that file. Up to 20 FC: entries may be specified for each main term or related term. Each FIL and CXT is 1-3 character entry.
See Reference	'USE: Term Name;' This entry specifies a preferred term which is to be substituted for the index term following the NT: entry. Only one USE: may be entered for a term.
Related Term	'RT: Term Name;' Up to 40 related terms may be entered for one index term. None may be entered if 'USE' has been specified.
See Also	'SA: Term Name;' Up to 40 see also terms may be entered for one index term. None may be entered if a 'USE' term was specified.

3. Delete Cards

Columns 1 - 4	Card Sequence Number
Columns 5 - 10	'DELETE' (indicates that the information which follows is to be deleted from an existing dictionary record)
Column 11	Blank

Columns 12 - 80	As described below (DELETE card entries have the same restrictions as do the ADD card entries except that the FC: entries may only follow the main term):
Function	PD 'F: (CD;SD;ID);' TD
Dictionary	'DICT: (DCT;DCT;...);'
See Reference	'USE: Term Name;'
Related Term	'RT: Term Name;'
See Also	'SA: Term Name;'
File Context	'FC: FIL-CXT;'
Scope	'SCOPE: ALL or $N_1;N_2;...N_m$' where N is a 1-3 digit number from 0 - 255. Up to 256 note numbers may be specified. If SCOPE is specified, it must be the last entry in the DELETE data set.

4. Replace cards

Columns 1 - 4	Card Sequence Number
Columns 5 - 11	'REPLACE'
Column 12	Blank
Columns 13 - 80	As described below:
See Reference	'USE : Term to be replaced; Replacing Term;'
Related Term	'RT: Term to be replaced; Replacing Term;'
See Also	'SA: Term to be replaced; Replacing Term;'

5. Scope cards

Columns 1 - 4	Card Sequence Number
Columns 5 - 9	'SCOPE'
Columns 10 - 12	Scope note line number Must be a three digit number 001 - 255 with any necessary leading zeros.
Column 13	Blank
Columns 14 - 73	Scope Note (all characters are valid)
Columns 74 - 80	Blank

An example showing the addition of a new term to the dictionary and the revision of an existing one is given below.

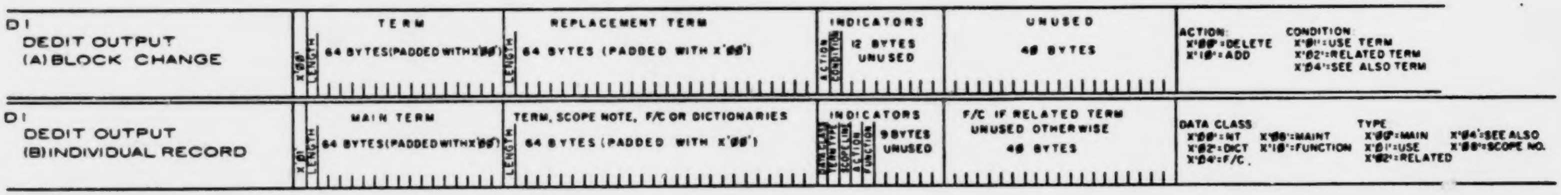
```
0001          HDR
0002          ADD NT:  EMPIRIN; F: (PD); DICT:(DRG);
0003               FC: HUM-DRG; RT:ACETYLSALICYLIC ACID
0006          SCOPE040 REF/AMERICAN DRUG INDEX
0007          HDR MEASURIN
0008          ADD DICT: (DRG); RT:NON-NARCOTIC
```

Card 0001 is a header card. It is blank from column 8 because it precedes the addition of a new main term to the dictionary. The term itself is EMPIRIN, shown following the code NT: in the first ADD card after the HDR. Card 0003 is a continuation of the data about EMPIRIN, giving the File/Context code and a related term. A single line of scope note is given for this term as well. Note that the line number need not be 001. It may be any number desired and will be stored as that line number. Thus, certain line numbers may be reserved for certain kinds of information, if desired.

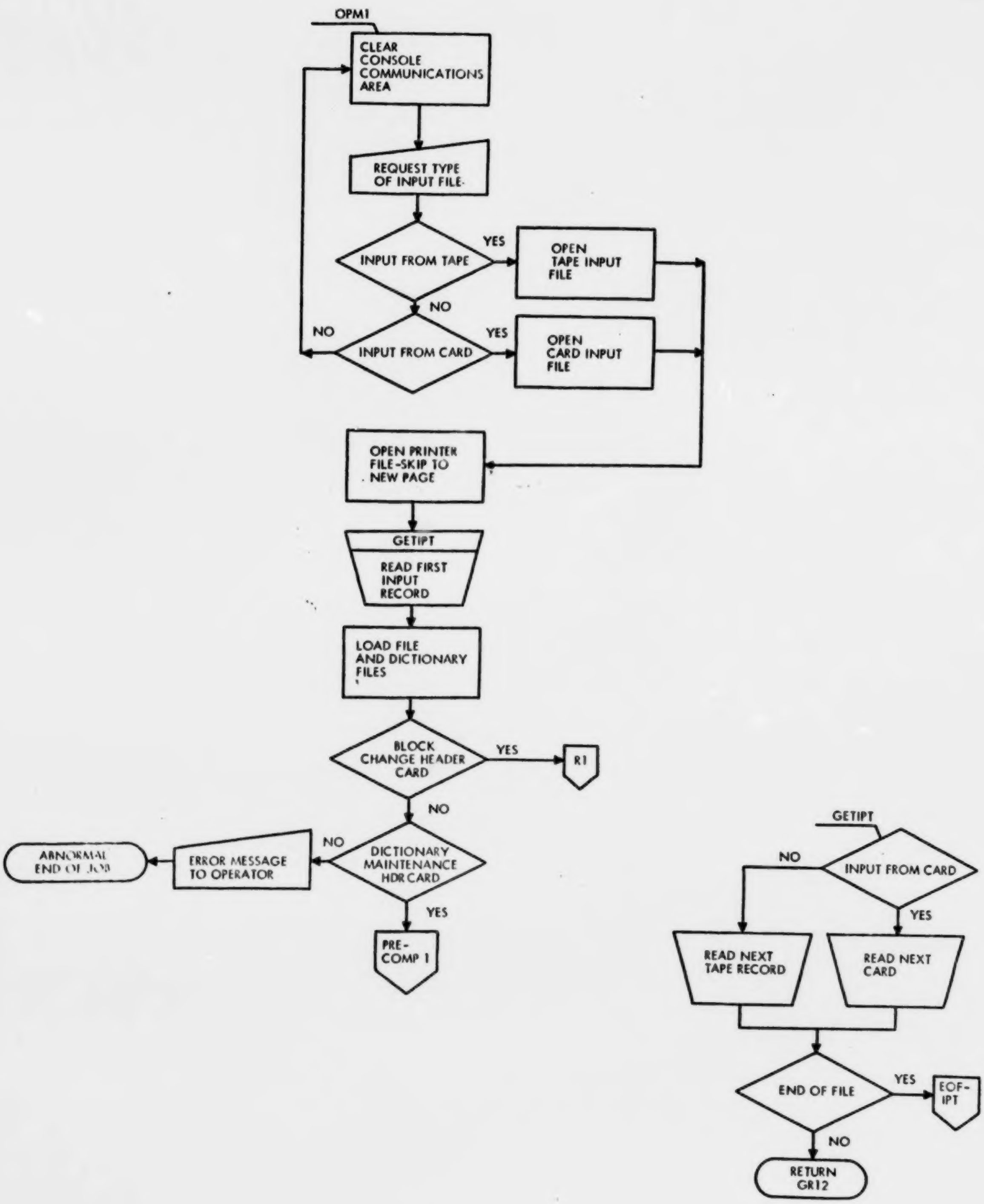
Output

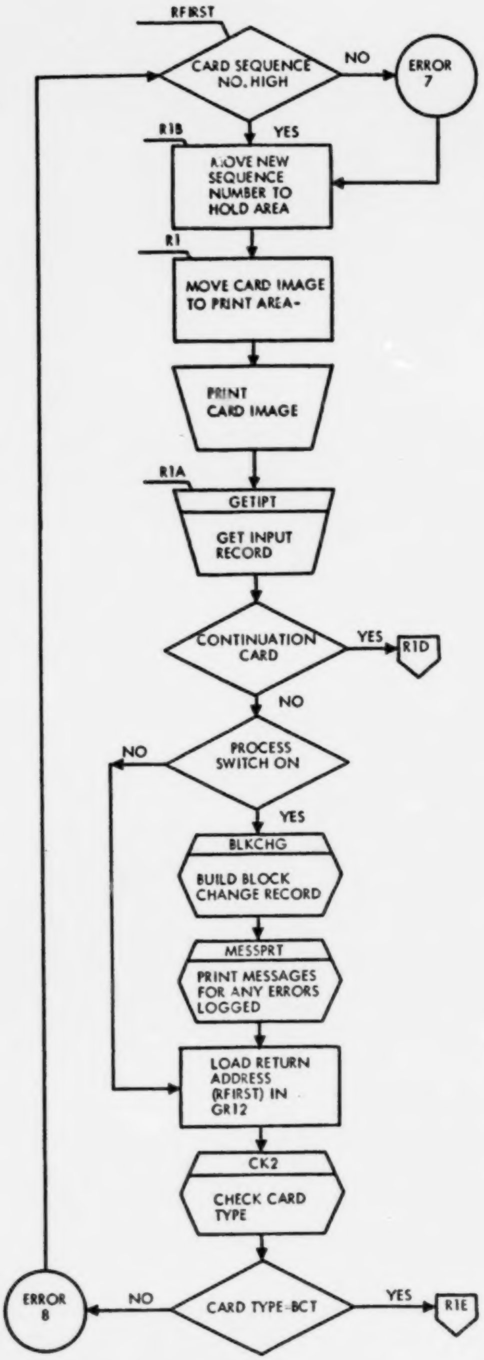
The output tape produced by DEDIT will be in the format shown below labeled D1. This tape, when sorted, will provide input to the Dictionary Maintenance Program (DICT2).

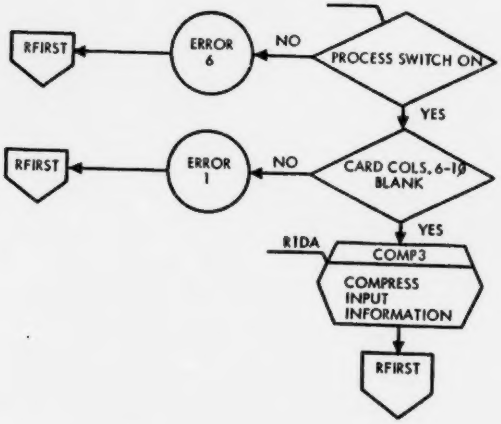
DICTIONARY EDIT (DEDIT) RECORD FORMAT

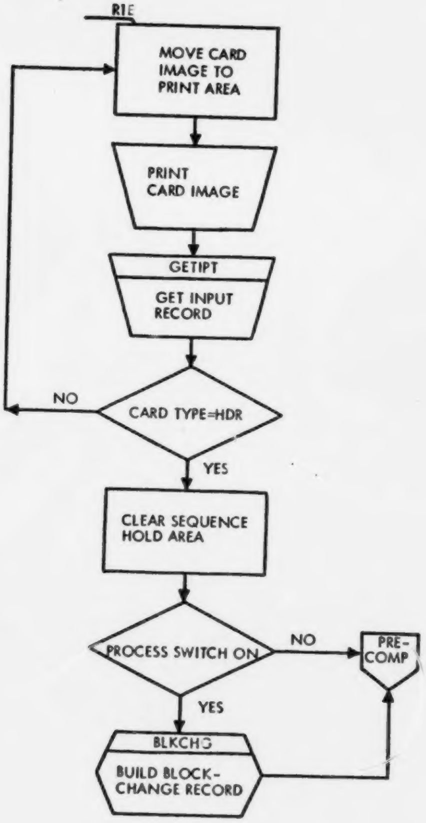


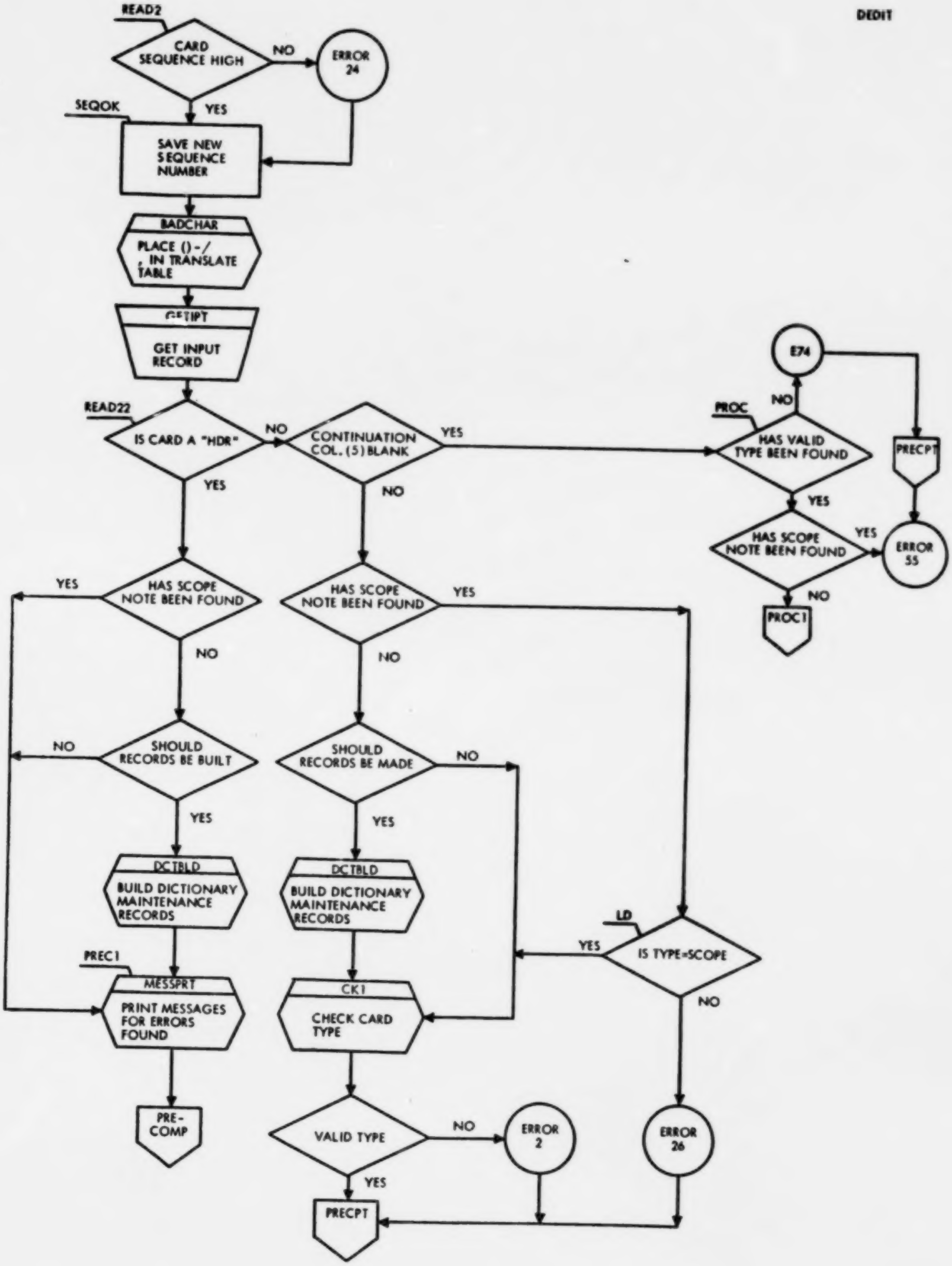
D1 DEDIT OUTPUT (A) BLOCK CHANGE	X'00'	LENGTH	TERM 64 BYTES (PADDED WITH X'00')	LENGTH	REPLACEMENT TERM 64 BYTES (PADDED WITH X'00')	INDICATORS ACTION CONDITION 12 BYTES UNUSED	UNUSED 40 BYTES	ACTION: X'00':DELETE X'10':ADD CONDITION: X'01':USE TERM X'02':RELATED TERM X'04':SEE ALSO TERM
D1 DEDIT OUTPUT (B) INDIVIDUAL RECORD	X'01'	LENGTH	MAIN TERM 64 BYTES (PADDED WITH X'00')	LENGTH	TERM, SCOPE NOTE, F/C OR DICTIONARIES 64 BYTES (PADDED WITH X'00')	INDICATORS DATA CLASS TERM TYPE SCOPE LINE ACTION FUNCTION 9 BYTES UNUSED	F/C IF RELATED TERM UNUSED OTHERWISE 40 BYTES	DATA CLASS X'00':NT X'02':DICT X'04':F/C. X'08':MAINT X'10':FUNCTION TYPE X'00':MAIN X'01':USE X'02':RELATED X'04':SEE ALSO X'08':SCOPE NO.

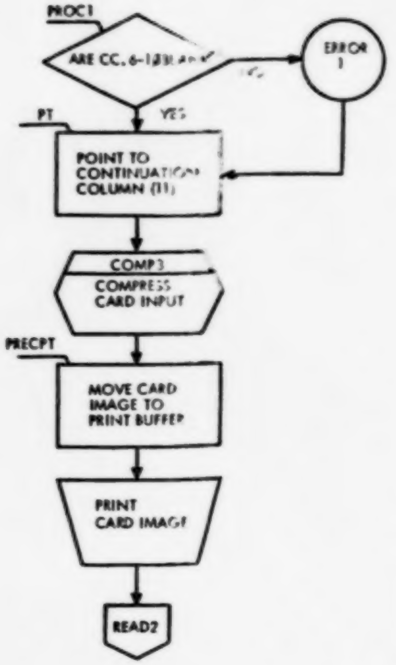

OPM1
CLEAR CONSOLE COMMUNICATIONS AREA
REQUEST TYPE OF INPUT FILE
INPUT FROM TAPE
YES
OPEN TAPE INPUT FILE
NO
INPUT FROM CARD
NO
YES
OPEN CARD INPUT FILE
OPEN PRINTER FILE-SKIP TO NEW PAGE
GETIPT
READ FIRST INPUT RECORD
LOAD FILE AND DICTIONARY FILES
BLOCK CHANGE HEADER CARD
YES
R1
NO
DICTIONARY MAINTENANCE HDR CARD
NO
ERROR MESSAGE TO OPERATOR
ABNORMAL END OF JOB
YES
PRE-COMP 1
GETIPT
INPUT FROM CARD
NO
YES
READ NEXT TAPE RECORD
READ NEXT CARD
END OF FILE
YES
EOF-IPT
NO
RETURN GR12

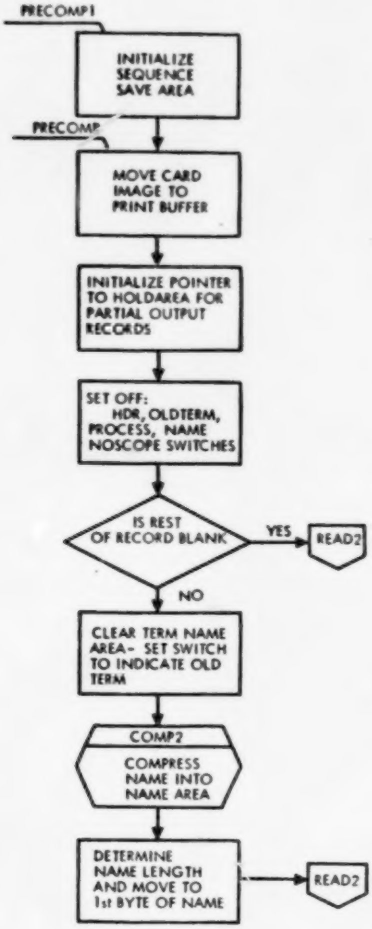

RFIRST
CARD SEQUENCE NO. HIGH
NO
ERROR 7
YES
R1B
MOVE NEW SEQUENCE NUMBER TO HOLD AREA
R1
MOVE CARD IMAGE TO PRINT AREA-
PRINT CARD IMAGE
R1A
GETIPT
GET INPUT RECORD
CONTINUATION CARD
YES
R1D
NO
PROCESS SWITCH ON
NO
YES
BLKCHG
BUILD BLOCK CHANGE RECORD
MESSPRT
PRINT MESSAGES FOR ANY ERRORS LOGGED
LOAD RETURN ADDRESS (RFIRST) IN GR12
CK2
CHECK CARD TYPE
ERROR 8
NO
CARD TYPE-BCT
YES
R1E

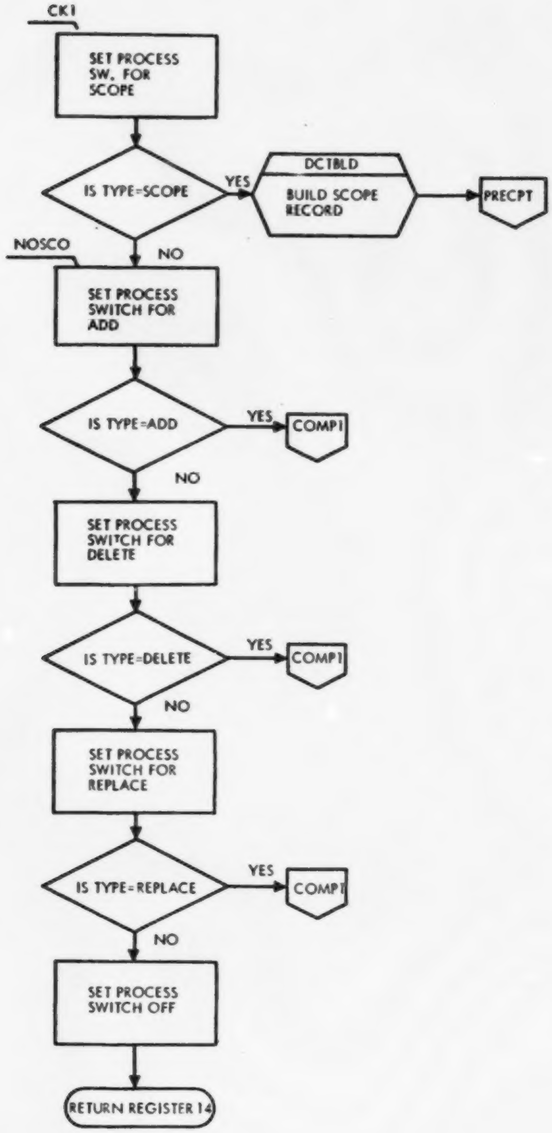

R1D
PROCESS SWITCH ON
NO
ERROR 6
RFIRST
YES
CARD COLS. 6-10 BLANK
NO
ERROR 1
RFIRST
YES
R1DA
COMP3
COMPRESS INPUT INFORMATION
RFIRST

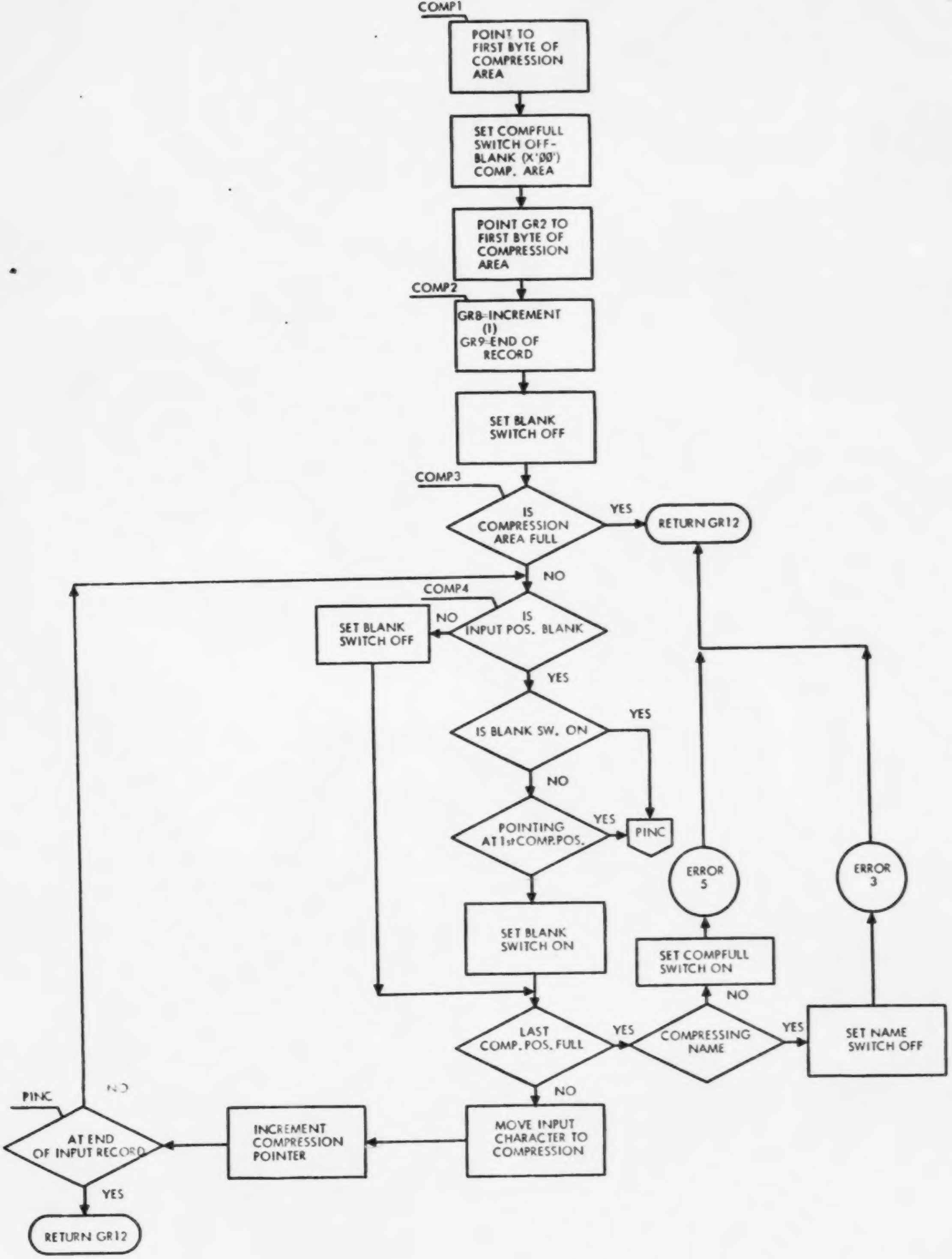

R1E
MOVE CARD IMAGE TO PRINT AREA
PRINT CARD IMAGE
GETIPT
GET INPUT RECORD
CARD TYPE=HDR
NO
YES
CLEAR SEQUENCE HOLD AREA
PROCESS SWITCH ON
NO
PRE-COMP
YES
BLKCHG
BUILD BLOCK-CHANGE RECORD

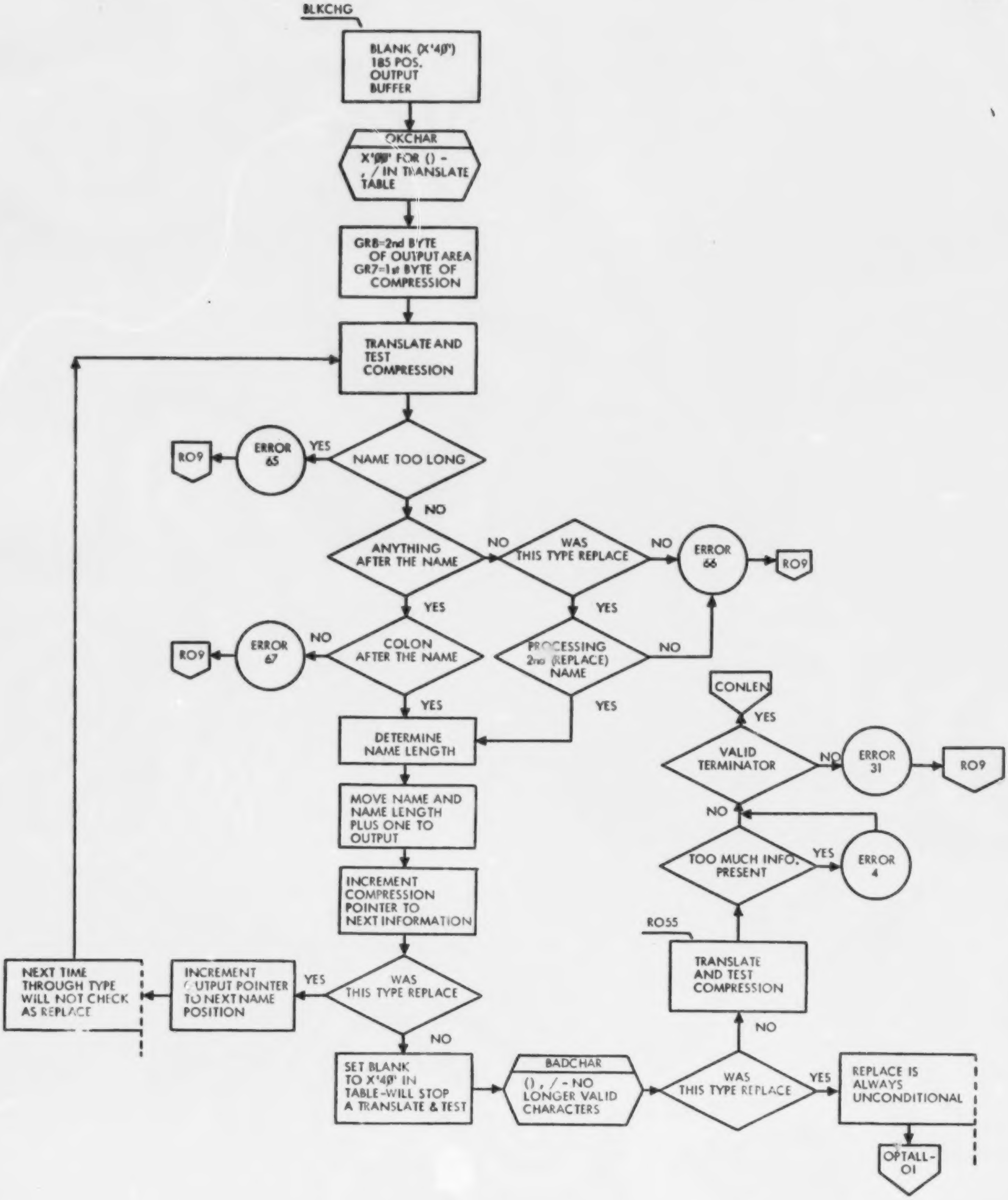

READ2
CARD SEQUENCE HIGH
NO
ERROR 24
YES
SEQOK
SAVE NEW SEQUENCE NUMBER
BADCHAR
PLACE ()-/ , IN TRANSLATE TABLE
GETIPT
GET INPUT RECORD
READ22
IS CARD A "HDR"
NO
CONTINUATION COL. (5) BLANK
YES
E74
NO
PROC
HAS VALID TYPE BEEN FOUND
PRECPT
YES
YES
NO
HAS SCOPE NOTE BEEN FOUND
YES
ERROR 55
NO
PROC1
YES
HAS SCOPE NOTE BEEN FOUND
NO
HAS SCOPE NOTE BEEN FOUND
YES
NO
NO
SHOULD RECORDS BE BUILT
YES
SHOULD RECORDS BE MADE
NO
YES
DCTBLD
BUILD DICTIONARY MAINTENANCE RECORDS
DCTBLD
BUILD DICTIONARY MAINTENANCE RECORDS
LD
YES
IS TYPE=SCOPE
NO
PREC1
MESSPRT
PRINT MESSAGES FOR ERRORS FOUND
CK1
CHECK CARD TYPE
PRE-COMP
VALID TYPE
NO
ERROR 2
ERROR 26
YES
PRECPT

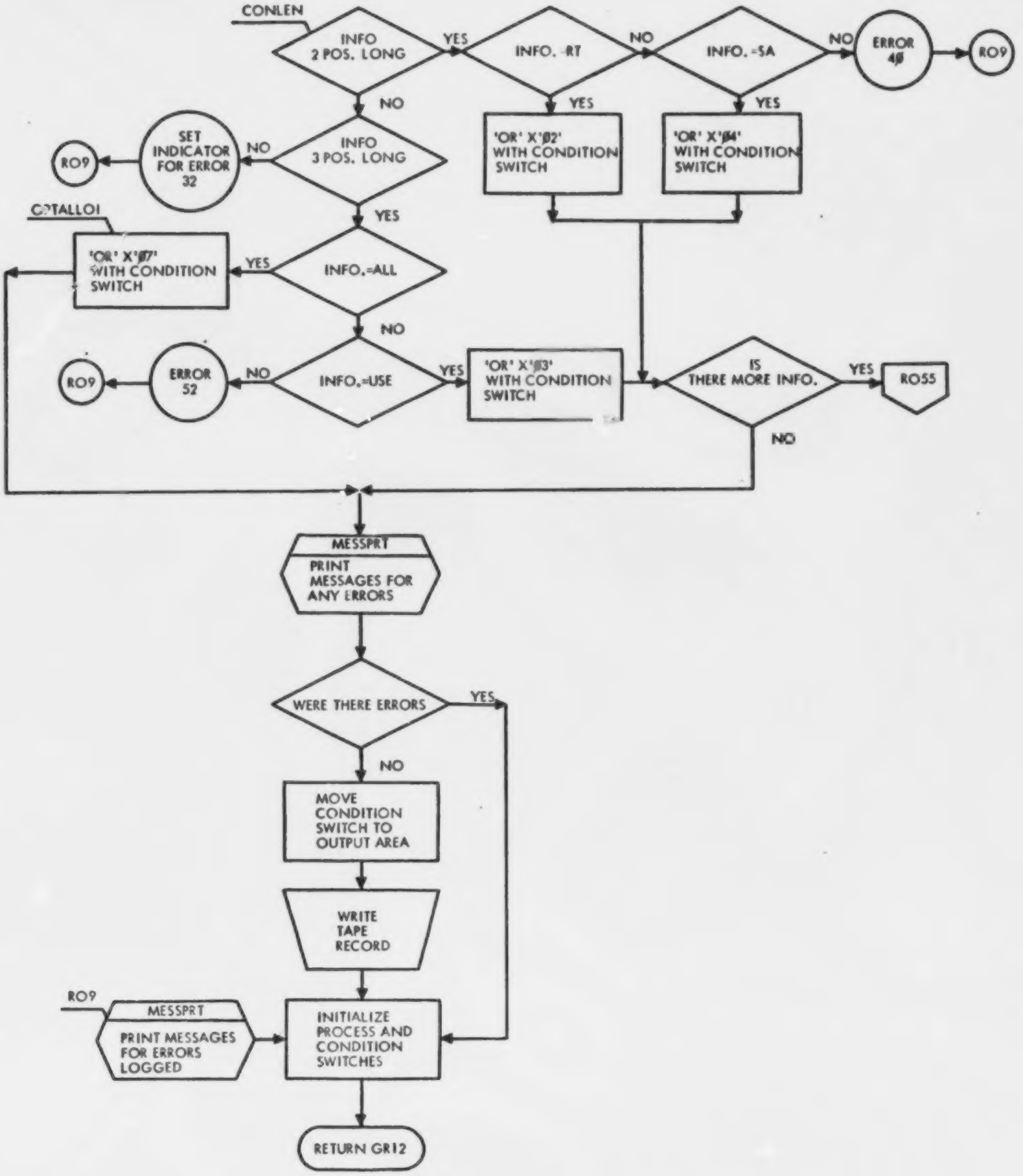

PROC1
ARE CC. 6-10 BLANK
NO
ERROR 1
YES
PT
POINT TO CONTINUATION COLUMN (11)
COMP3
COMPRESS CARD INPUT
PRECPT
MOVE CARD IMAGE TO PRINT BUFFER
PRINT CARD IMAGE
READ2

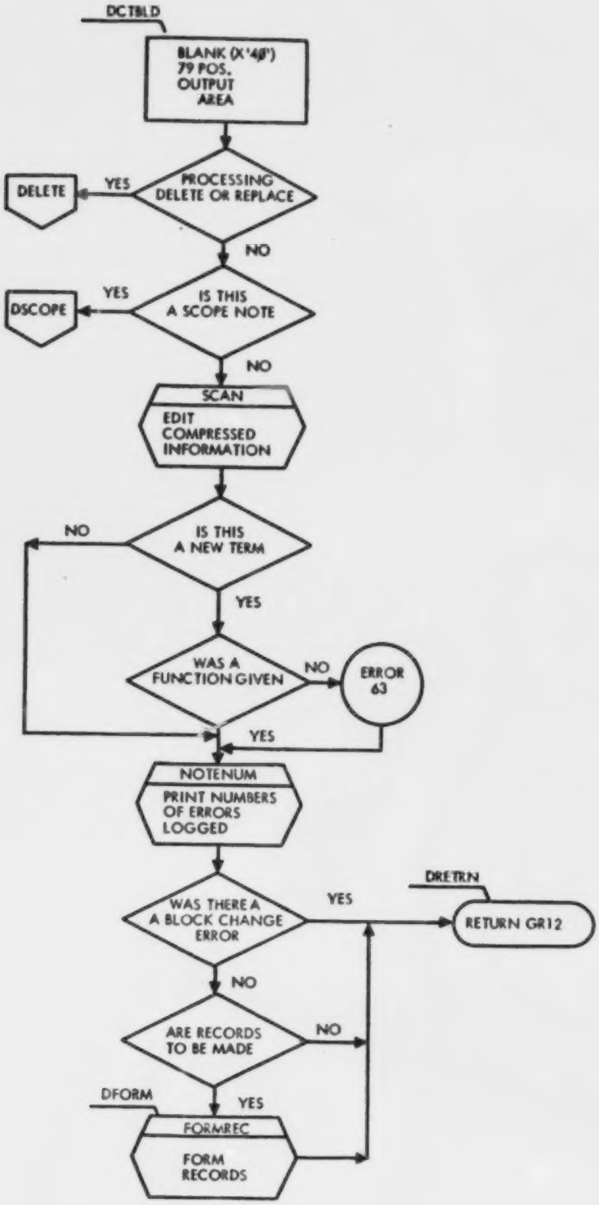

PRECOMP1
INITIALIZE SEQUENCE SAVE AREA
PRECOMP
MOVE CARD IMAGE TO PRINT BUFFER
INITIALIZE POINTER TO HOLDAREA FOR PARTIAL OUTPUT RECORDS
SET OFF: HDR, OLDTERM, PROCESS, NAME NOSCOPE SWITCHES
IS REST OF RECORD BLANK
YES
READ2
NO
CLEAR TERM NAME AREA- SET SWITCH TO INDICATE OLD TERM
COMP2
COMPRESS NAME INTO NAME AREA
DETERMINE NAME LENGTH AND MOVE TO 1st BYTE OF NAME
READ2

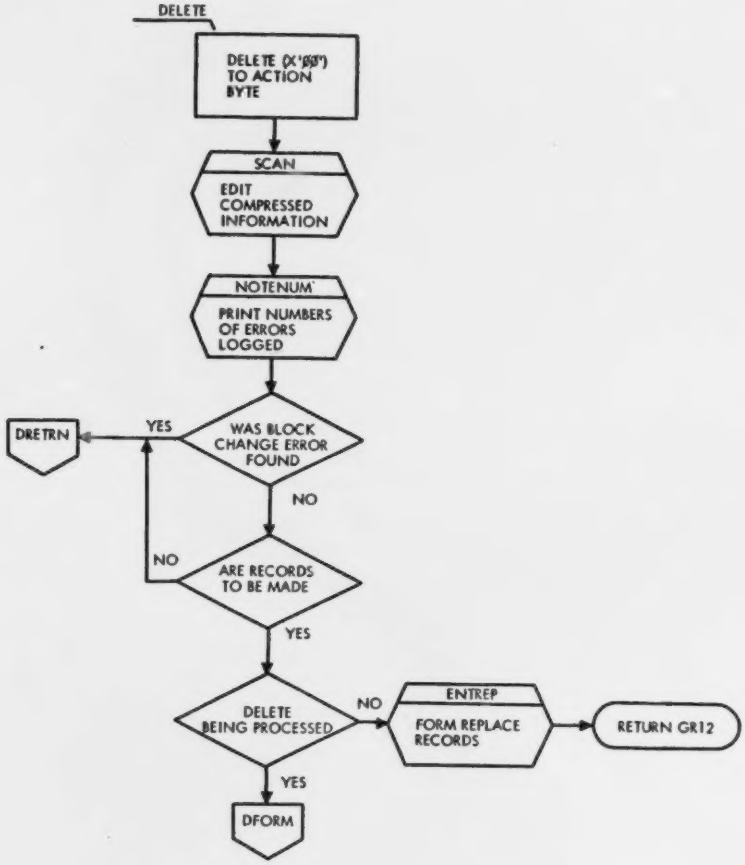

CK1
SET PROCESS SW. FOR SCOPE
IS TYPE=SCOPE
YES
DCTBLD
BUILD SCOPE RECORD
PRECPT
NO
NOSCO
SET PROCESS SWITCH FOR ADD
IS TYPE=ADD
YES
COMP1
NO
SET PROCESS SWITCH FOR DELETE
IS TYPE=DELETE
YES
COMP1
NO
SET PROCESS SWITCH FOR REPLACE
IS TYPE=REPLACE
YES
COMP1
NO
SET PROCESS SWITCH OFF
RETURN REGISTER 14

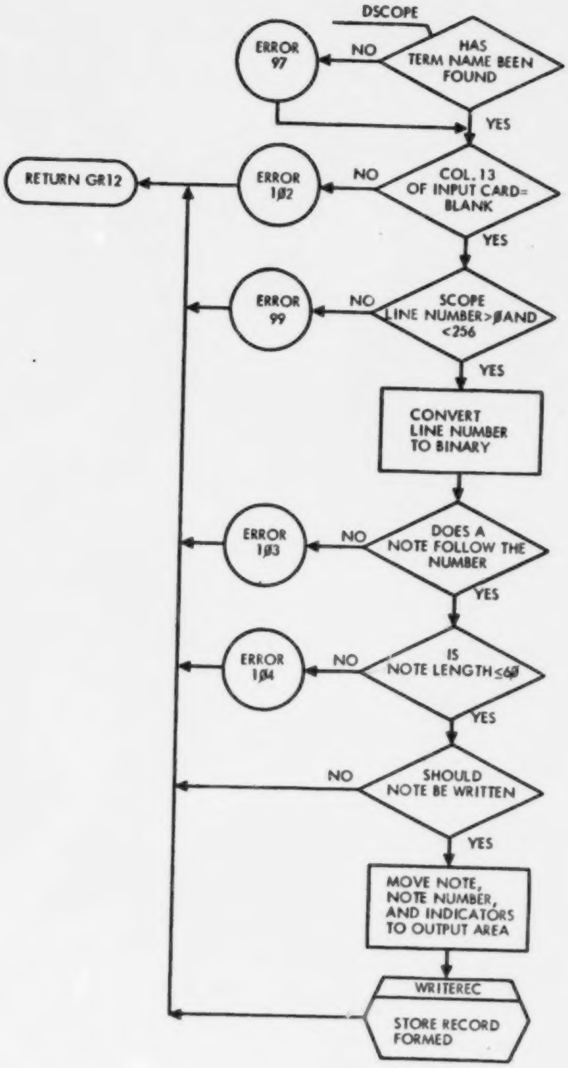

COMP1
POINT TO FIRST BYTE OF COMPRESSION AREA
SET COMPFULL SWITCH OFF-BLANK (X'00') COMP. AREA
POINT GR2 TO FIRST BYTE OF COMPRESSION AREA
COMP2
GR8=INCREMENT (1) GR9=END OF RECORD
SET BLANK SWITCH OFF
COMP3
IS COMPRESSION AREA FULL
YES
RETURN GR12
NO
COMP4
IS INPUT POS. BLANK
NO
SET BLANK SWITCH OFF
YES
IS BLANK SW. ON
YES
NO
POINTING AT 1st COMP.POS.
YES
PINC
ERROR 5
ERROR 3
SET BLANK SWITCH ON
SET COMPFULL SWITCH ON
NO
LAST COMP. POS. FULL
YES
COMPRESSING NAME
YES
SET NAME SWITCH OFF
NO
NO
PINC
AT END OF INPUT RECORD
INCREMENT COMPRESSION POINTER
MOVE INPUT CHARACTER TO COMPRESSION
YES
RETURN GR12

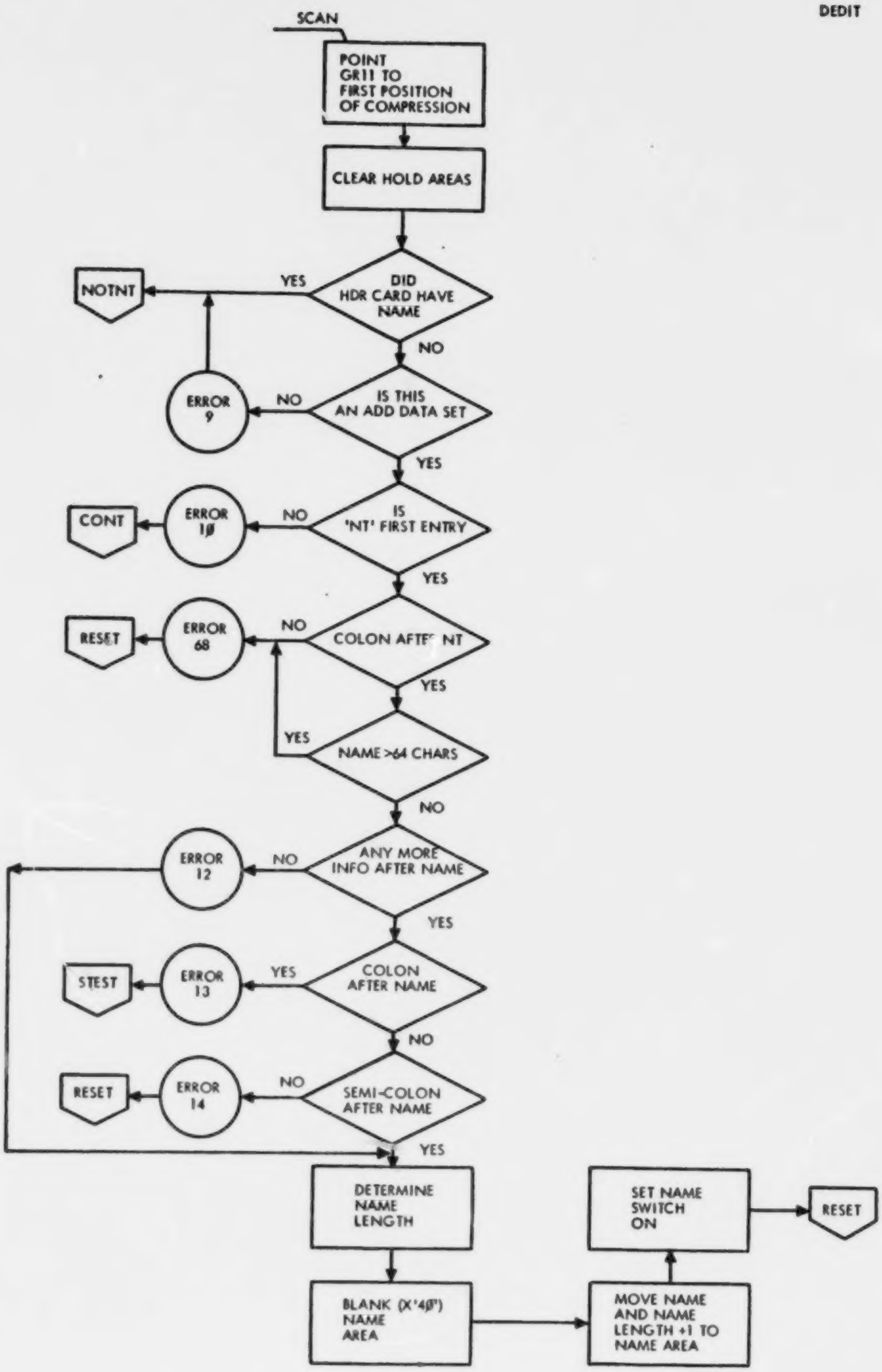

BLKCHG
BLANK (X'40') 185 POS. OUTPUT BUFFER
OKCHAR
X'00' FOR () - , / IN TRANSLATE TABLE
GR8=2nd BYTE OF OUTPUT AREA GR7=1st BYTE OF COMPRESSION
TRANSLATE AND TEST COMPRESSION
NAME TOO LONG
YES
ERROR 65
RO9
NO
ANYTHING AFTER THE NAME
NO
WAS THIS TYPE REPLACE
NO
ERROR 66
RO9
YES
YES
COLON AFTER THE NAME
NO
ERROR 67
RO9
PROCESSING 2nd (REPLACE) NAME
NO
YES
YES
DETERMINE NAME LENGTH
MOVE NAME AND NAME LENGTH PLUS ONE TO OUTPUT
INCREMENT COMPRESSION POINTER TO NEXT INFORMATION
WAS THIS TYPE REPLACE
YES
INCREMENT OUTPUT POINTER TO NEXT NAME POSITION
NEXT TIME THROUGH TYPE WILL NOT CHECK AS REPLACE
NO
SET BLANK TO X'40' IN TABLE-WILL STOP A TRANSLATE & TEST
BADCHAR
() , / - NO LONGER VALID CHARACTERS
WAS THIS TYPE REPLACE
YES
REPLACE IS ALWAYS UNCONDITIONAL
OPTALL-01
NO
RO55
TRANSLATE AND TEST COMPRESSION
TOO MUCH INFO. PRESENT
YES
ERROR 4
NO
VALID TERMINATOR
NO
ERROR 31
RO9
YES
CONLEN


CONLEN
INFO 2 POS. LONG
YES
INFO. =RT
NO
INFO. =SA
NO
ERROR 40
RO9
NO
YES
YES
'OR' X'02' WITH CONDITION SWITCH
'OR' X'04' WITH CONDITION SWITCH
RO9
SET INDICATOR FOR ERROR 32
NO
INFO 3 POS. LONG
YES
CPTALL01
'OR' X'07' WITH CONDITION SWITCH
YES
INFO.=ALL
NO
RO9
ERROR 52
NO
INFO.=USE
YES
'OR' X'03' WITH CONDITION SWITCH
IS THERE MORE INFO.
YES
RO55
NO
MESSPRT
PRINT MESSAGES FOR ANY ERRORS
WERE THERE ERRORS
YES
NO
MOVE CONDITION SWITCH TO OUTPUT AREA
WRITE TAPE RECORD
RO9
MESSPRT
PRINT MESSAGES FOR ERRORS LOGGED
INITIALIZE PROCESS AND CONDITION SWITCHES
RETURN GR12

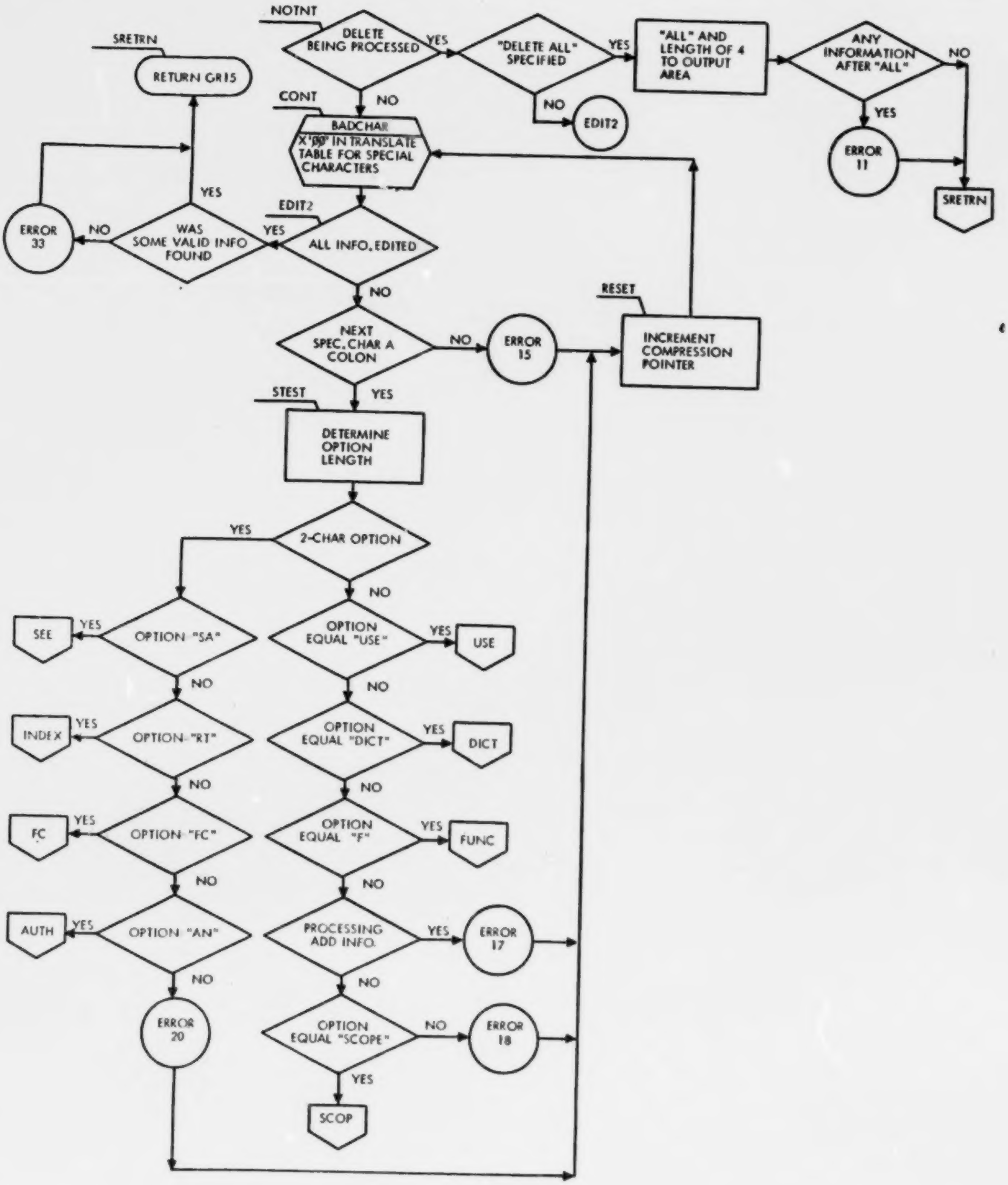

DCTBLD
BLANK (X'40') 79 POS. OUTPUT AREA
PROCESSING DELETE OR REPLACE
YES
DELETE
NO
IS THIS A SCOPE NOTE
YES
DSCOPE
NO
SCAN
EDIT COMPRESSED INFORMATION
IS THIS A NEW TERM
NO
YES
WAS A FUNCTION GIVEN
NO
ERROR 63
YES
NOTENUM
PRINT NUMBERS OF ERRORS LOGGED
WAS THERE A A BLOCK CHANGE ERROR
YES
DRETRN
RETURN GR12
NO
ARE RECORDS TO BE MADE
NO
YES
DFORM
FORMREC
FORM RECORDS

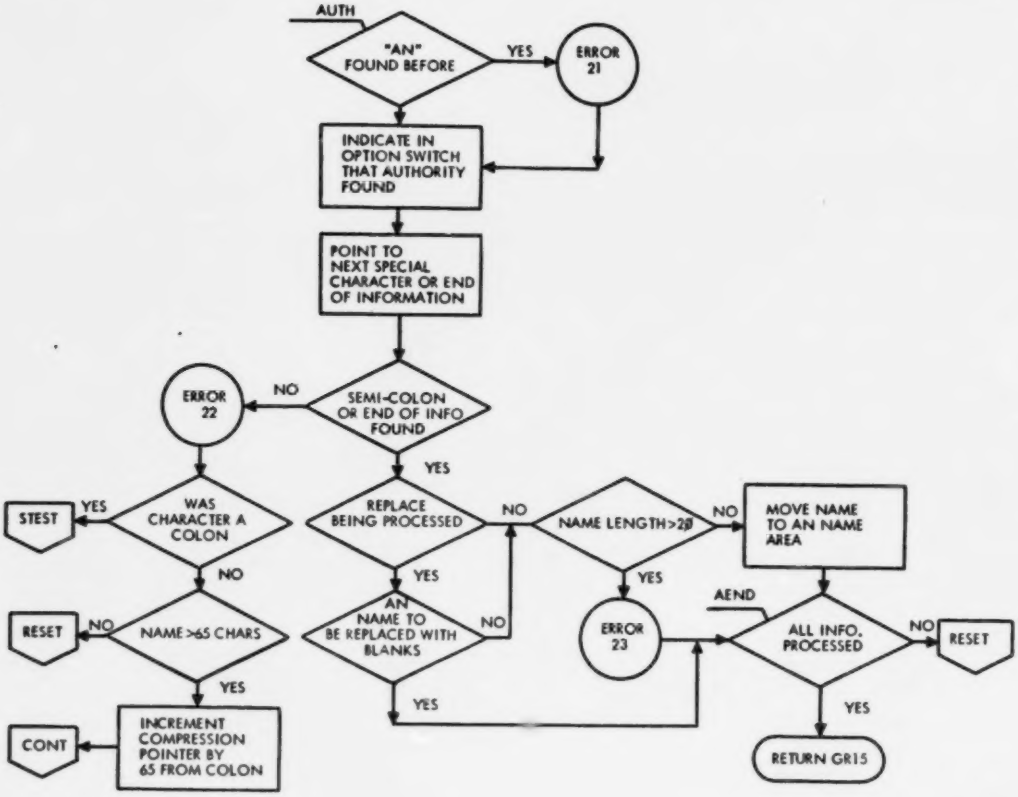

DELETE
DELETE (X'00') TO ACTION BYTE
SCAN
EDIT COMPRESSED INFORMATION
NOTENUM
PRINT NUMBERS OF ERRORS LOGGED
WAS BLOCK CHANGE ERROR FOUND
YES
DRETRN
NO
ARE RECORDS TO BE MADE
NO
YES
DELETE BEING PROCESSED
NO
ENTREP
FORM REPLACE RECORDS
RETURN GR12
YES
DFORM

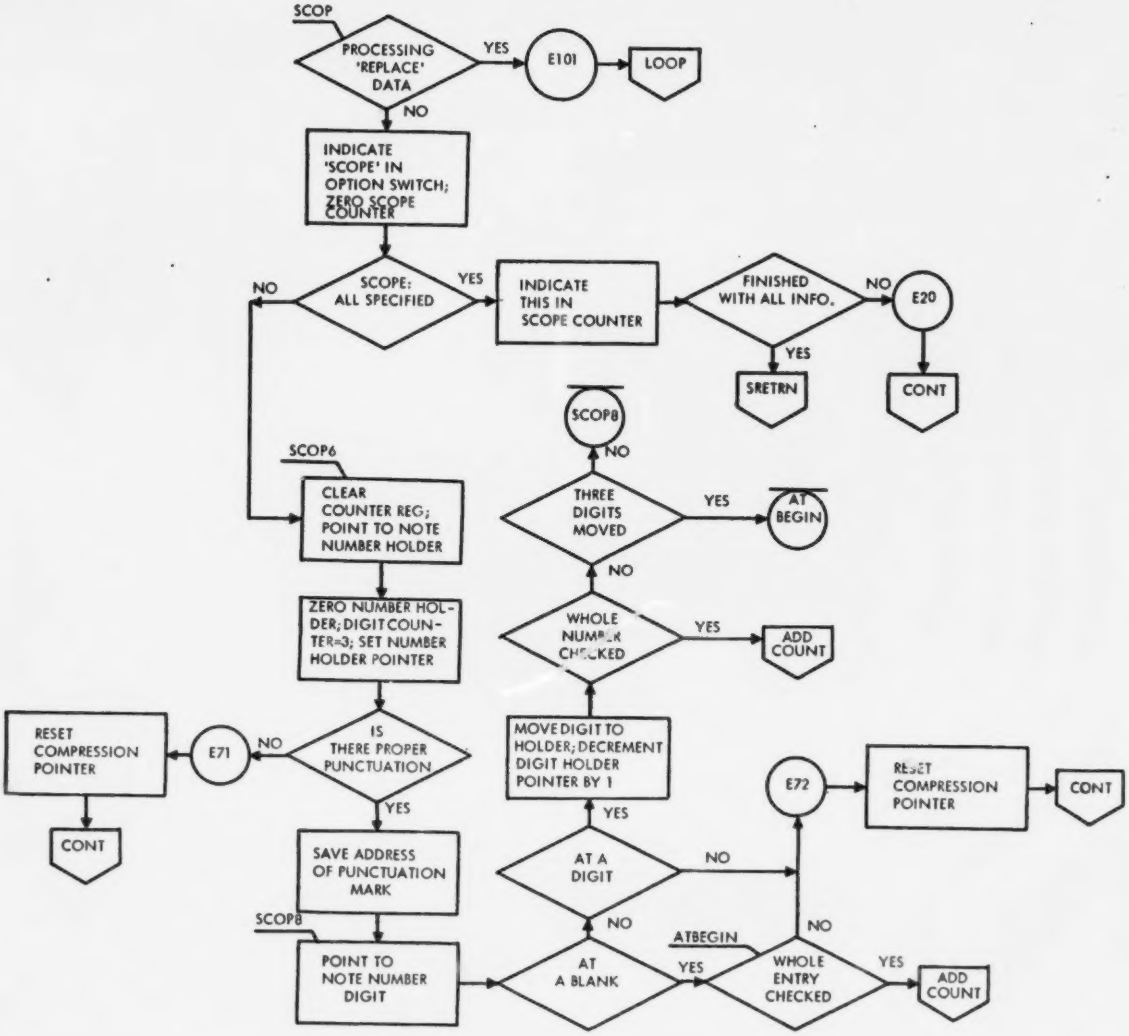

DSCOPE
HAS TERM NAME BEEN FOUND
NO
ERROR 97
YES
COL. 13 OF INPUT CARD= BLANK
NO
ERROR 102
RETURN GR12
YES
SCOPE LINE NUMBER>0 AND <256
NO
ERROR 99
YES
CONVERT LINE NUMBER TO BINARY
DOES A NOTE FOLLOW THE NUMBER
NO
ERROR 103
YES
IS NOTE LENGTH≤60
NO
ERROR 104
YES
SHOULD NOTE BE WRITTEN
NO
YES
MOVE NOTE, NOTE NUMBER, AND INDICATORS TO OUTPUT AREA
WRITEREC
STORE RECORD FORMED

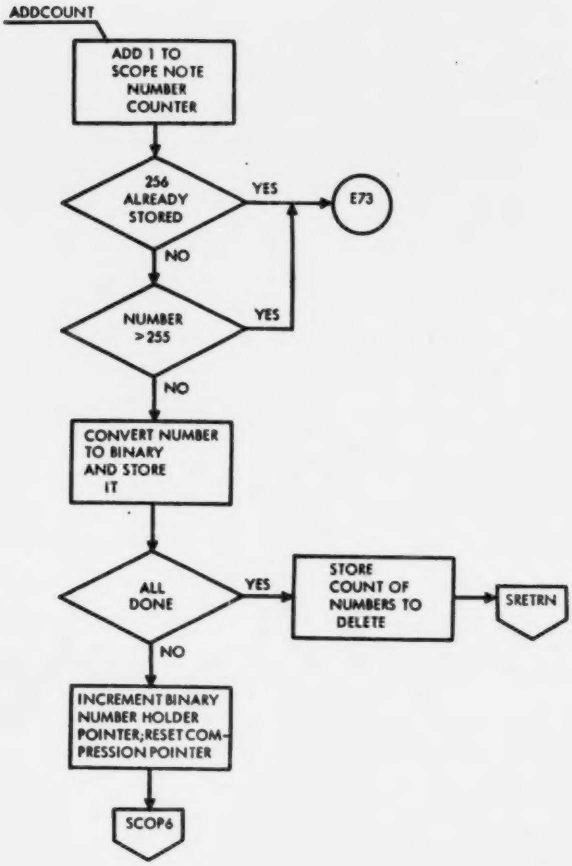

SCAN
POINT GR11 TO FIRST POSITION OF COMPRESSION
CLEAR HOLD AREAS
DID HDR CARD HAVE NAME
YES
NOTNT
NO
IS THIS AN ADD DATA SET
NO
ERROR 9
YES
IS 'NT' FIRST ENTRY
NO
ERROR 10
CONT
YES
COLON AFTER NT
NO
ERROR 68
RESET
YES
NAME >64 CHARS
YES
NO
ANY MORE INFO AFTER NAME
NO
ERROR 12
YES
COLON AFTER NAME
YES
ERROR 13
STEST
NO
SEMI-COLON AFTER NAME
NO
ERROR 14
RESET
YES
DETERMINE NAME LENGTH
BLANK (X'40') NAME AREA
MOVE NAME AND NAME LENGTH +1 TO NAME AREA
SET NAME SWITCH ON
RESET

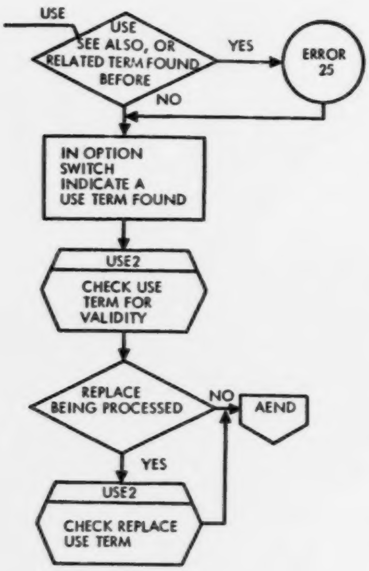
DEDIT

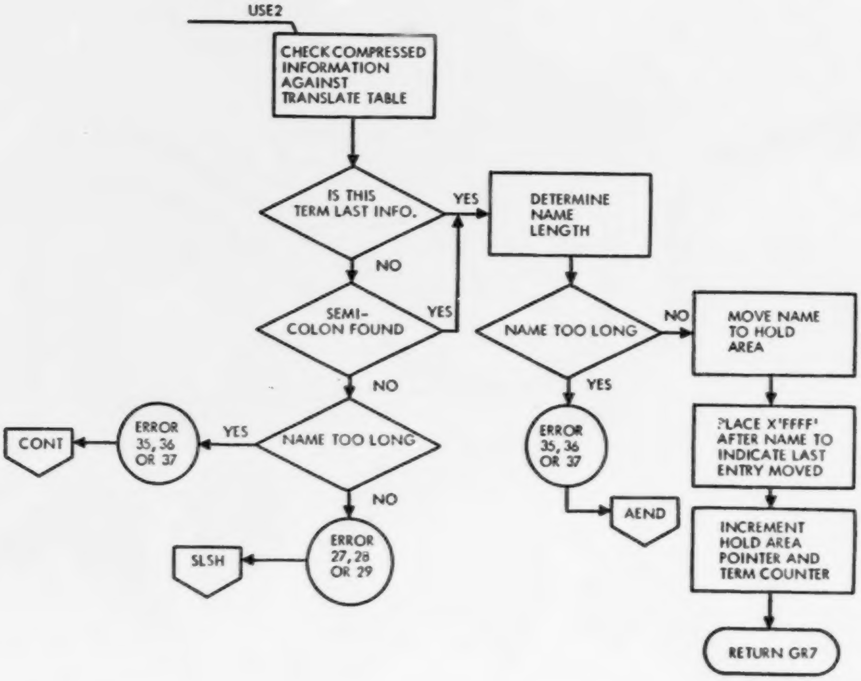

NOTNT
DELETE BEING PROCESSED
YES
"DELETE ALL" SPECIFIED
YES
"ALL" AND LENGTH OF 4 TO OUTPUT AREA
ANY INFORMATION AFTER "ALL"
NO
YES
ERROR 11
SRETRN
NO
EDIT2
SRETRN
RETURN GR15
NO
CONT
BADCHAR
X'00' IN TRANSLATE TABLE FOR SPECIAL CHARACTERS
YES
EDIT2
ERROR 33
NO
WAS SOME VALID INFO FOUND
YES
ALL INFO. EDITED
NO
RESET
NEXT SPEC. CHAR A COLON
NO
ERROR 15
INCREMENT COMPRESSION POINTER
STEST
YES
DETERMINE OPTION LENGTH
YES
2-CHAR OPTION
NO
SEE
YES
OPTION "SA"
OPTION EQUAL "USE"
YES
USE
NO
NO
INDEX
YES
OPTION "RT"
OPTION EQUAL "DICT"
YES
DICT
NO
NO
FC
YES
OPTION "FC"
OPTION EQUAL "F"
YES
FUNC
NO
NO
AUTH
YES
OPTION "AN"
PROCESSING ADD INFO.
YES
ERROR 17
NO
NO
ERROR 20
OPTION EQUAL "SCOPE"
NO
ERROR 18
YES
SCOP

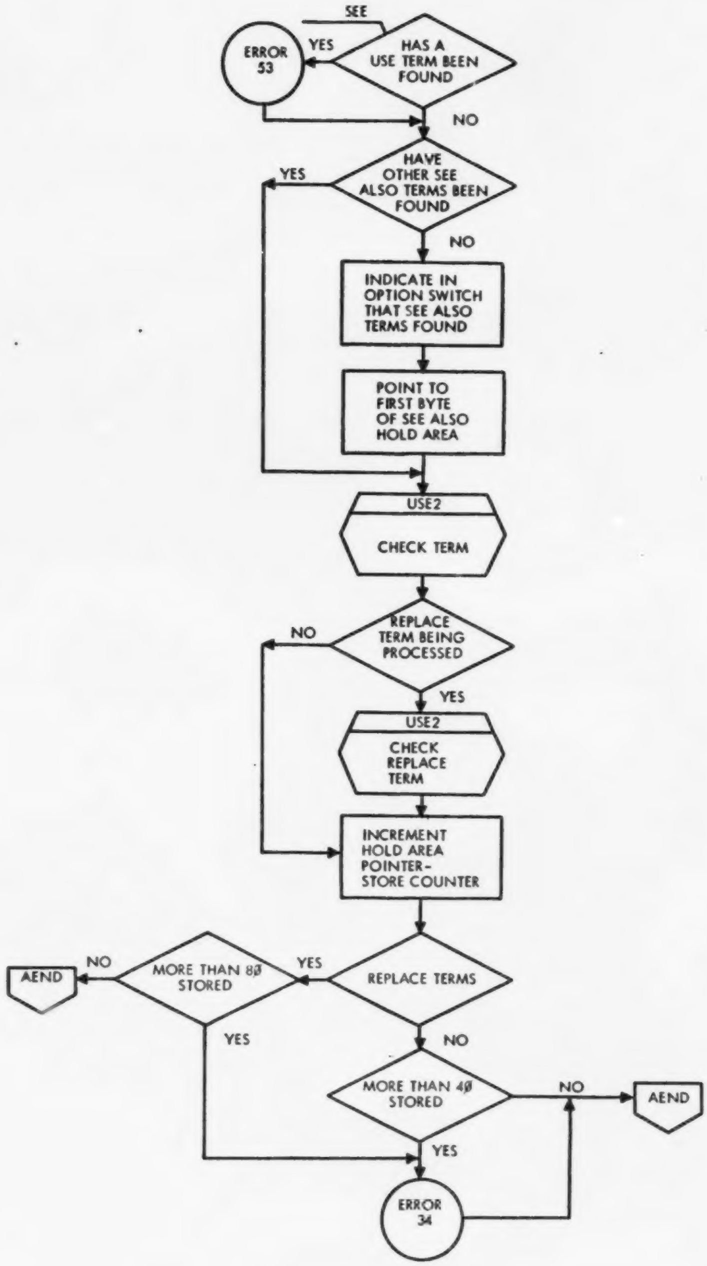

AUTH
"AN" FOUND BEFORE
YES
ERROR 21
INDICATE IN OPTION SWITCH THAT AUTHORITY FOUND
POINT TO NEXT SPECIAL CHARACTER OR END OF INFORMATION
SEMI-COLON OR END OF INFO FOUND
NO
ERROR 22
YES
WAS CHARACTER A COLON
YES
STEST
NO
NAME >65 CHARS
NO
RESET
YES
INCREMENT COMPRESSION POINTER BY 65 FROM COLON
CONT
REPLACE BEING PROCESSED
NO
YES
AN NAME TO BE REPLACED WITH BLANKS
NO
YES
NAME LENGTH>20
NO
MOVE NAME TO AN NAME AREA
YES
ERROR 23
AEND
ALL INFO. PROCESSED
NO
RESET
YES
RETURN GR15

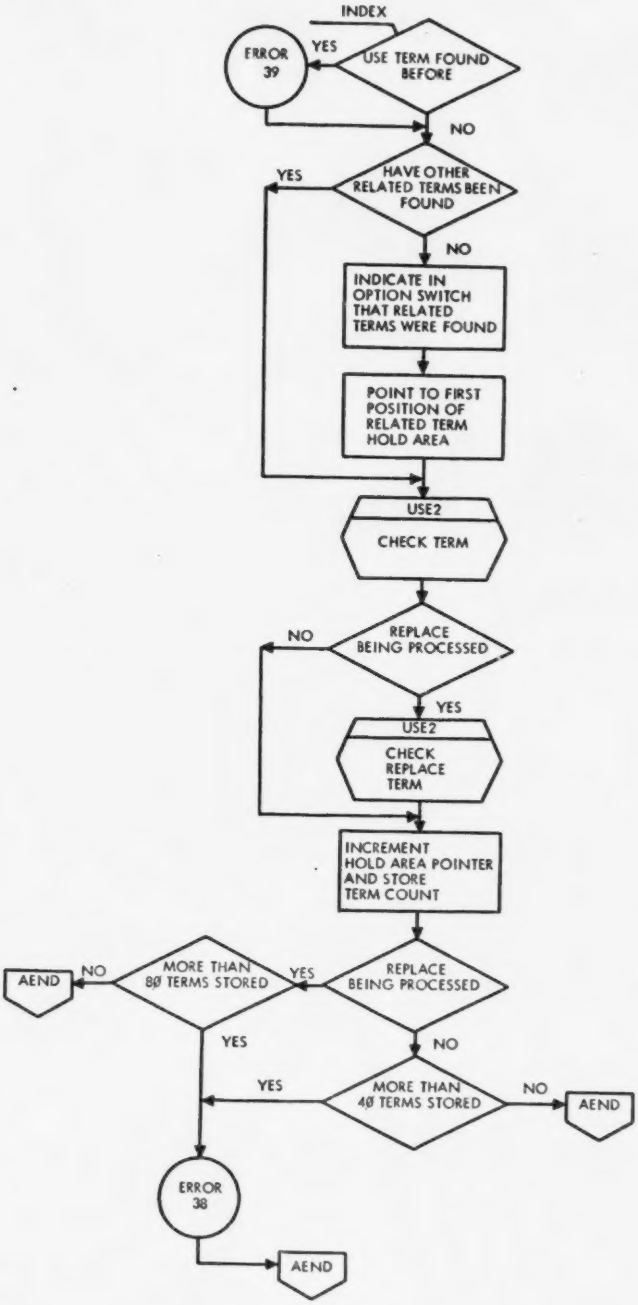

SCOP
PROCESSING 'REPLACE' DATA
YES
E101
LOOP
NO
INDICATE 'SCOPE' IN OPTION SWITCH; ZERO SCOPE COUNTER
NO
SCOPE: ALL SPECIFIED
YES
INDICATE THIS IN SCOPE COUNTER
FINISHED WITH ALL INFO.
NO
E20
YES
SRETRN
CONT
SCOP8
NO
SCOP6
CLEAR COUNTER REG; POINT TO NOTE NUMBER HOLDER
THREE DIGITS MOVED
YES
AT BEGIN
NO
ZERO NUMBER HOLDER; DIGIT COUNTER=3; SET NUMBER HOLDER POINTER
WHOLE NUMBER CHECKED
YES
ADD COUNT
MOVE DIGIT TO HOLDER; DECREMENT DIGIT HOLDER POINTER BY 1
RESET COMPRESSION POINTER
E71
NO
IS THERE PROPER PUNCTUATION
E72
RESET COMPRESSION POINTER
CONT
YES
YES
CONT
SAVE ADDRESS OF PUNCTUATION MARK
AT A DIGIT
NO
NO
SCOP8
NO
ATBEGIN
POINT TO NOTE NUMBER DIGIT
AT A BLANK
YES
WHOLE ENTRY CHECKED
YES
ADD COUNT

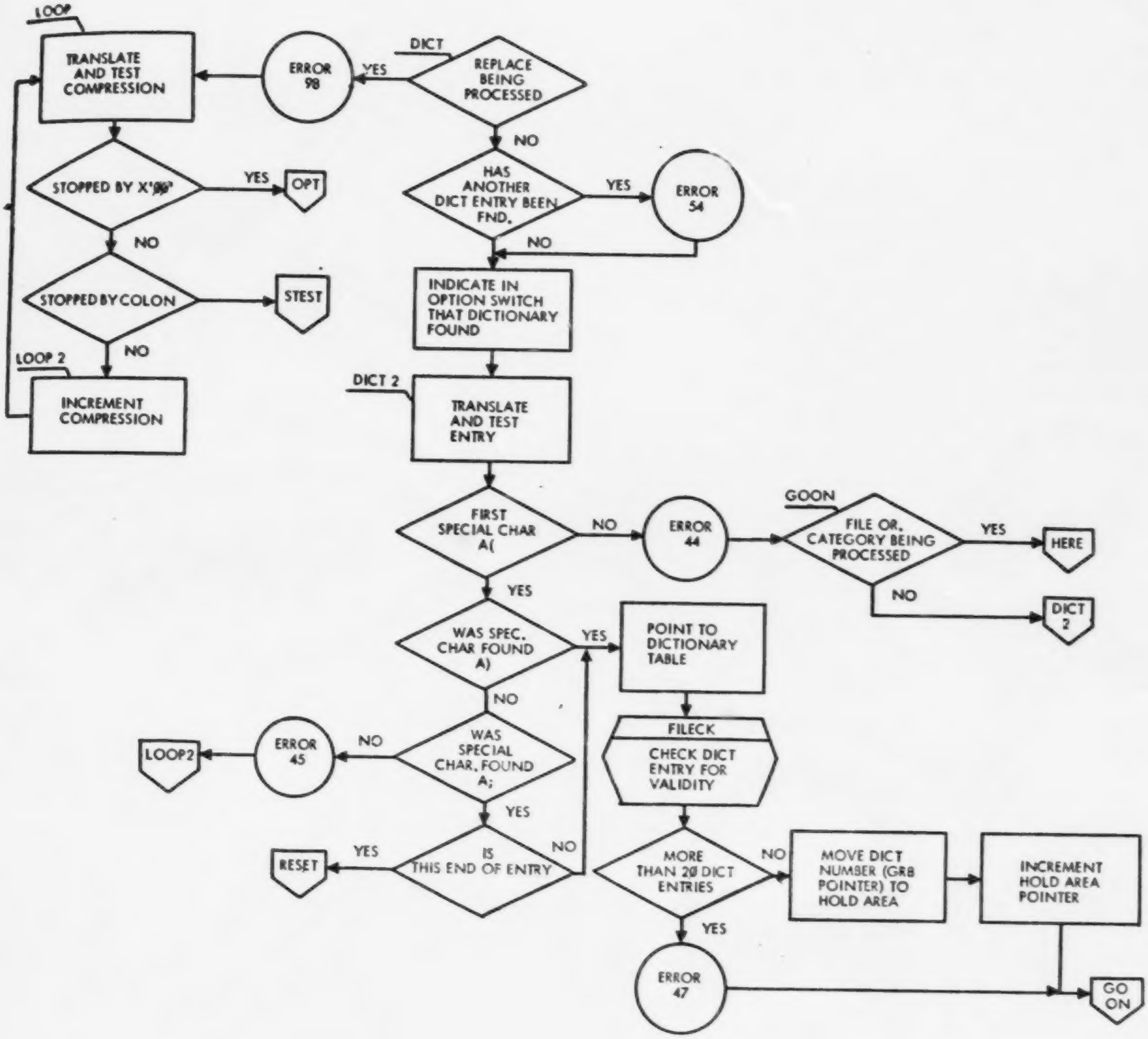

ADDCOUNT
ADD 1 TO SCOPE NOTE NUMBER COUNTER
256 ALREADY STORED
YES
E73
NO
NUMBER >255
YES
NO
CONVERT NUMBER TO BINARY AND STORE IT
ALL DONE
YES
STORE COUNT OF NUMBERS TO DELETE
SRETRN
NO
INCREMENT BINARY NUMBER HOLDER POINTER; RESET COMPRESSION POINTER
SCOP6

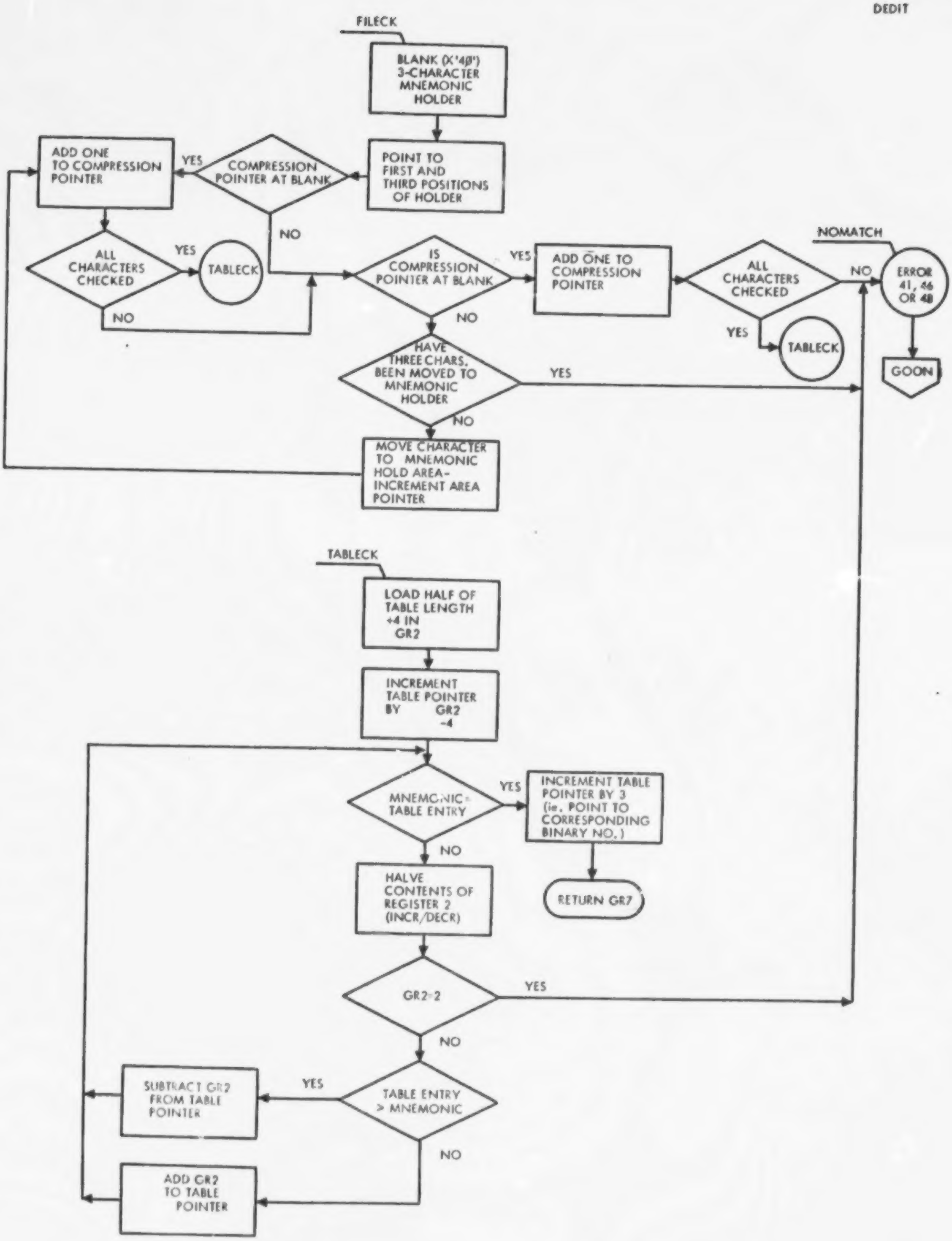

USE
USE SEE ALSO, OR RELATED TERM FOUND BEFORE
YES
ERROR 25
NO
IN OPTION SWITCH INDICATE A USE TERM FOUND
USE2
CHECK USE TERM FOR VALIDITY
REPLACE BEING PROCESSED
NO
AEND
YES
USE2
CHECK REPLACE USE TERM

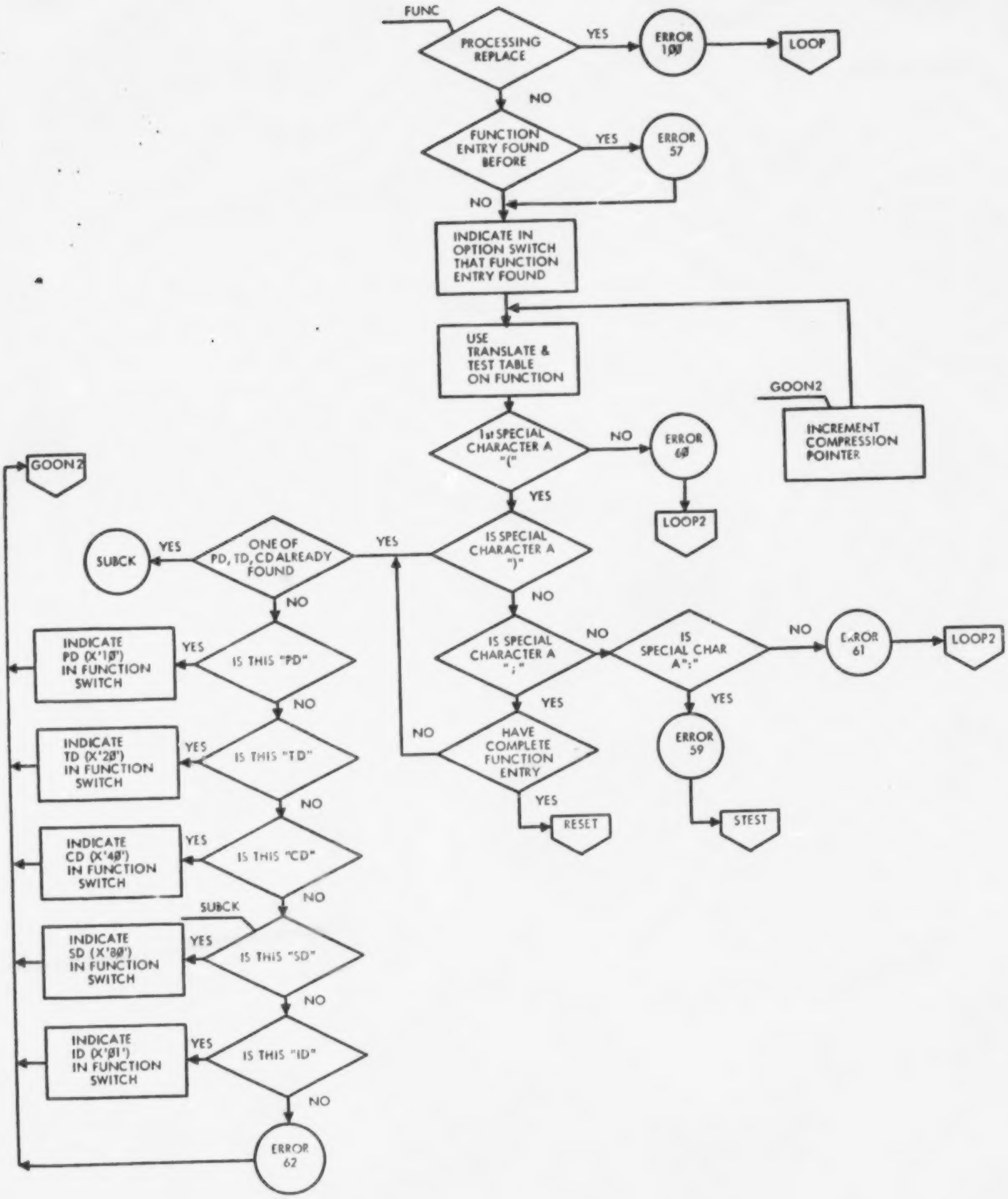

USE2
CHECK COMPRESSED INFORMATION AGAINST TRANSLATE TABLE
IS THIS TERM LAST INFO.
YES
DETERMINE NAME LENGTH
NO
SEMI-COLON FOUND
YES
NAME TOO LONG
NO
MOVE NAME TO HOLD AREA
NO
YES
PLACE X'FFFF' AFTER NAME TO INDICATE LAST ENTRY MOVED
CONT
ERROR 35, 36 OR 37
YES
NAME TOO LONG
ERROR 35, 36 OR 37
AEND
INCREMENT HOLD AREA POINTER AND TERM COUNTER
NO
SLSH
ERROR 27, 28 OR 29
RETURN GR7

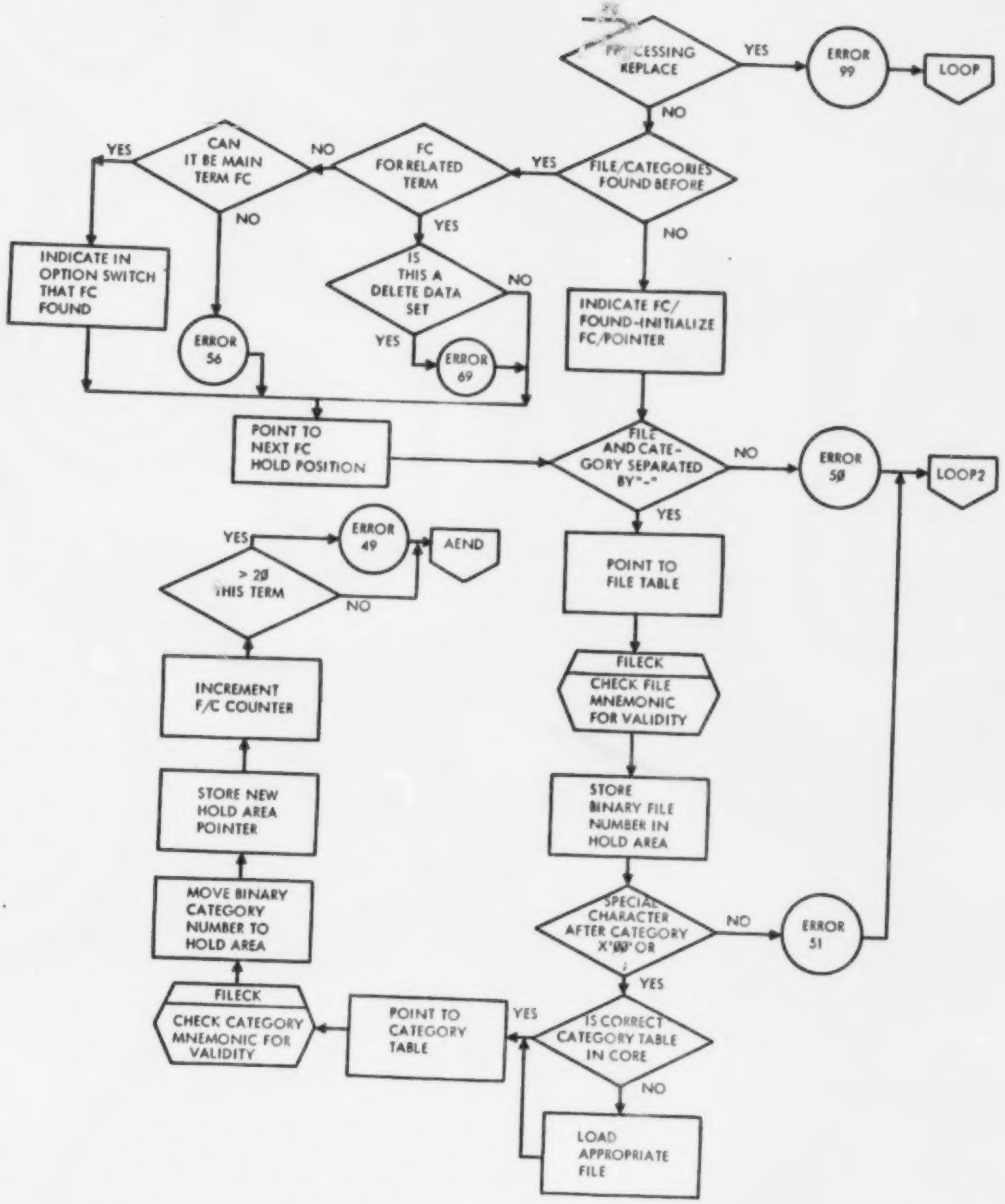

SEE
HAS A USE TERM BEEN FOUND
YES
ERROR 53
NO
HAVE OTHER SEE ALSO TERMS BEEN FOUND
YES
NO
INDICATE IN OPTION SWITCH THAT SEE ALSO TERMS FOUND
POINT TO FIRST BYTE OF SEE ALSO HOLD AREA
USE2
CHECK TERM
REPLACE TERM BEING PROCESSED
NO
YES
USE2
CHECK REPLACE TERM
INCREMENT HOLD AREA POINTER-STORE COUNTER
REPLACE TERMS
YES
MORE THAN 80 STORED
NO
AEND
YES
NO
MORE THAN 40 STORED
NO
AEND
YES
ERROR 34


INDEX
USE TERM FOUND BEFORE
YES
ERROR 39
NO
HAVE OTHER RELATED TERMS BEEN FOUND
YES
NO
INDICATE IN OPTION SWITCH THAT RELATED TERMS WERE FOUND
POINT TO FIRST POSITION OF RELATED TERM HOLD AREA
USE2
CHECK TERM
REPLACE BEING PROCESSED
NO
YES
USE2
CHECK REPLACE TERM
INCREMENT HOLD AREA POINTER AND STORE TERM COUNT
REPLACE BEING PROCESSED
YES
MORE THAN 80 TERMS STORED
NO
AEND
YES
NO
MORE THAN 40 TERMS STORED
YES
NO
AEND
ERROR 38
AEND

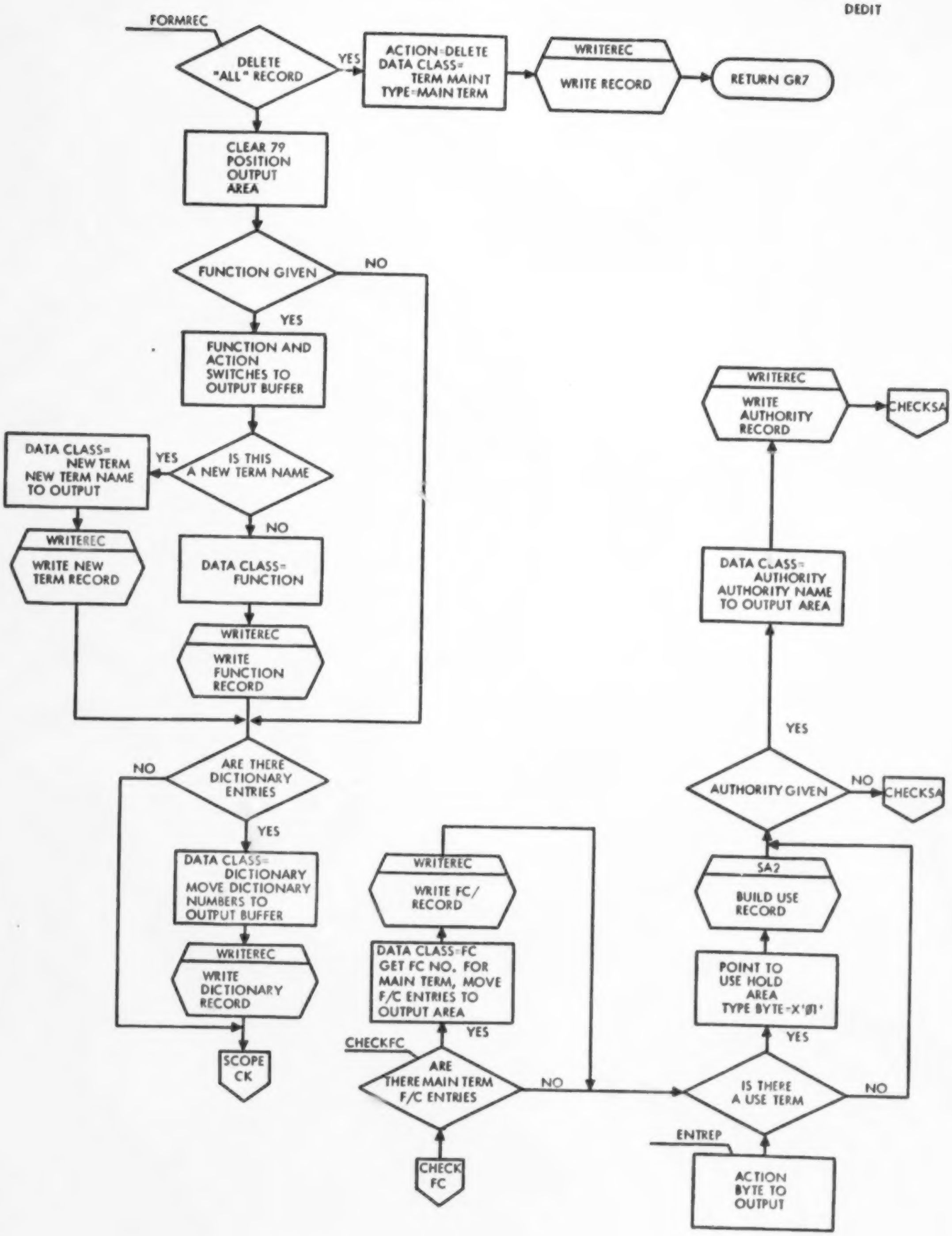

LOOP
TRANSLATE AND TEST COMPRESSION
ERROR 98
YES
DICT
REPLACE BEING PROCESSED
STOPPED BY X'00'
YES
OPT
NO
HAS ANOTHER DICT ENTRY BEEN FND.
YES
ERROR 54
NO
NO
STOPPED BY COLON
STEST
INDICATE IN OPTION SWITCH THAT DICTIONARY FOUND
NO
LOOP 2
INCREMENT COMPRESSION
DICT 2
TRANSLATE AND TEST ENTRY
GOON
FIRST SPECIAL CHAR A(
NO
ERROR 44
FILE OR. CATEGORY BEING PROCESSED
YES
HERE
NO
DICT 2
YES
WAS SPEC. CHAR FOUND A)
YES
POINT TO DICTIONARY TABLE
NO
FILECK
CHECK DICT ENTRY FOR VALIDITY
LOOP2
ERROR 45
NO
WAS SPECIAL CHAR. FOUND A;
YES
YES
RESET
IS THIS END OF ENTRY
NO
MORE THAN 20 DICT ENTRIES
NO
MOVE DICT NUMBER (GRB POINTER) TO HOLD AREA
INCREMENT HOLD AREA POINTER
YES
ERROR 47
GO ON

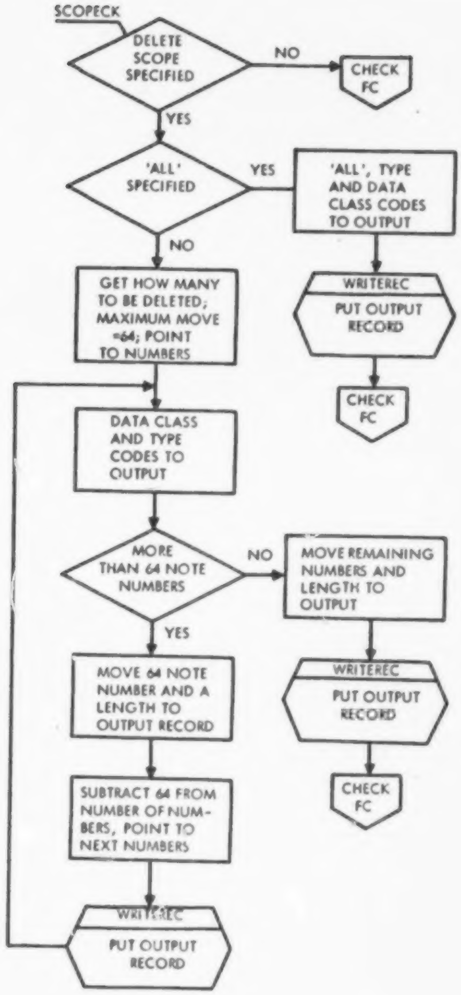

FILECK
BLANK (X'40')
3-CHARACTER
MNEMONIC
HOLDER
POINT TO
FIRST AND
THIRD POSITIONS
OF HOLDER
COMPRESSION
POINTER AT BLANK
YES
NO
ADD ONE
TO COMPRESSION
POINTER
ALL
CHARACTERS
CHECKED
YES
TABLECK
NO
IS
COMPRESSION
POINTER AT BLANK
YES
ADD ONE TO
COMPRESSION
POINTER
ALL
CHARACTERS
CHECKED
YES
TABLECK
NO
NOMATCH
ERROR
41, 46
OR 48
GOON
NO
HAVE
THREE CHARS.
BEEN MOVED TO
MNEMONIC
HOLDER
YES
NO
MOVE CHARACTER
TO MNEMONIC
HOLD AREA-
INCREMENT AREA
POINTER
TABLECK
LOAD HALF OF
TABLE LENGTH
+4 IN
GR2
INCREMENT
TABLE POINTER
BY GR2
-4
MNEMONIC=
TABLE ENTRY
YES
INCREMENT TABLE
POINTER BY 3
(ie. POINT TO
CORRESPONDING
BINARY NO.)
RETURN GR7
NO
HALVE
CONTENTS OF
REGISTER 2
(INCR/DECR)
GR2=2
YES
NO
TABLE ENTRY
> MNEMONIC
YES
SUBTRACT GR2
FROM TABLE
POINTER
NO
ADD GR2
TO TABLE
POINTER

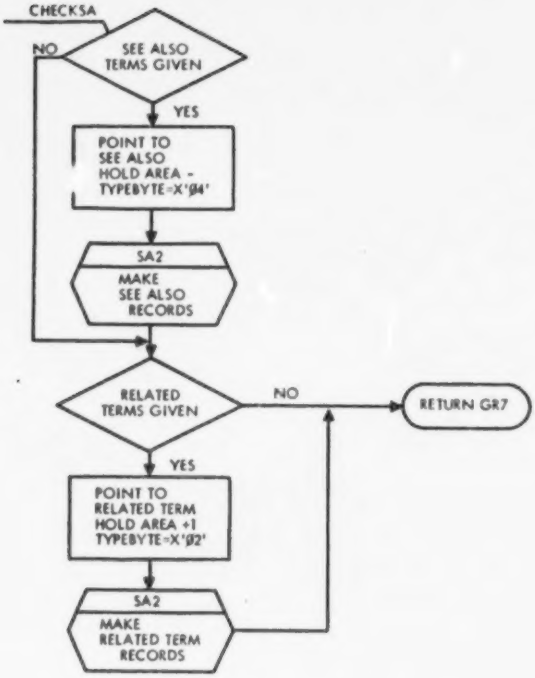

FUNC
PROCESSING REPLACE
YES
ERROR 100
LOOP
NO
FUNCTION ENTRY FOUND BEFORE
YES
ERROR 57
NO
INDICATE IN OPTION SWITCH THAT FUNCTION ENTRY FOUND
USE TRANSLATE & TEST TABLE ON FUNCTION
GOON2
INCREMENT COMPRESSION POINTER
1st SPECIAL CHARACTER A "("
NO
ERROR 60
LOOP2
YES
GOON2
SUBCK
YES
ONE OF PD, TD, CD ALREADY FOUND
YES
IS SPECIAL CHARACTER A ")"
NO
NO
INDICATE PD (X'10') IN FUNCTION SWITCH
YES
IS THIS "PD"
IS SPECIAL CHARACTER A ";"
NO
IS SPECIAL CHAR A ":"
NO
ERROR 61
LOOP2
NO
YES
YES
INDICATE TD (X'20') IN FUNCTION SWITCH
YES
IS THIS "TD"
NO
HAVE COMPLETE FUNCTION ENTRY
ERROR 59
NO
YES
INDICATE CD (X'40') IN FUNCTION SWITCH
YES
IS THIS "CD"
RESET
STEST
NO
SUBCK
INDICATE SD (X'80') IN FUNCTION SWITCH
YES
IS THIS "SD"
NO
INDICATE ID (X'01') IN FUNCTION SWITCH
YES
IS THIS "ID"
NO
ERROR 62

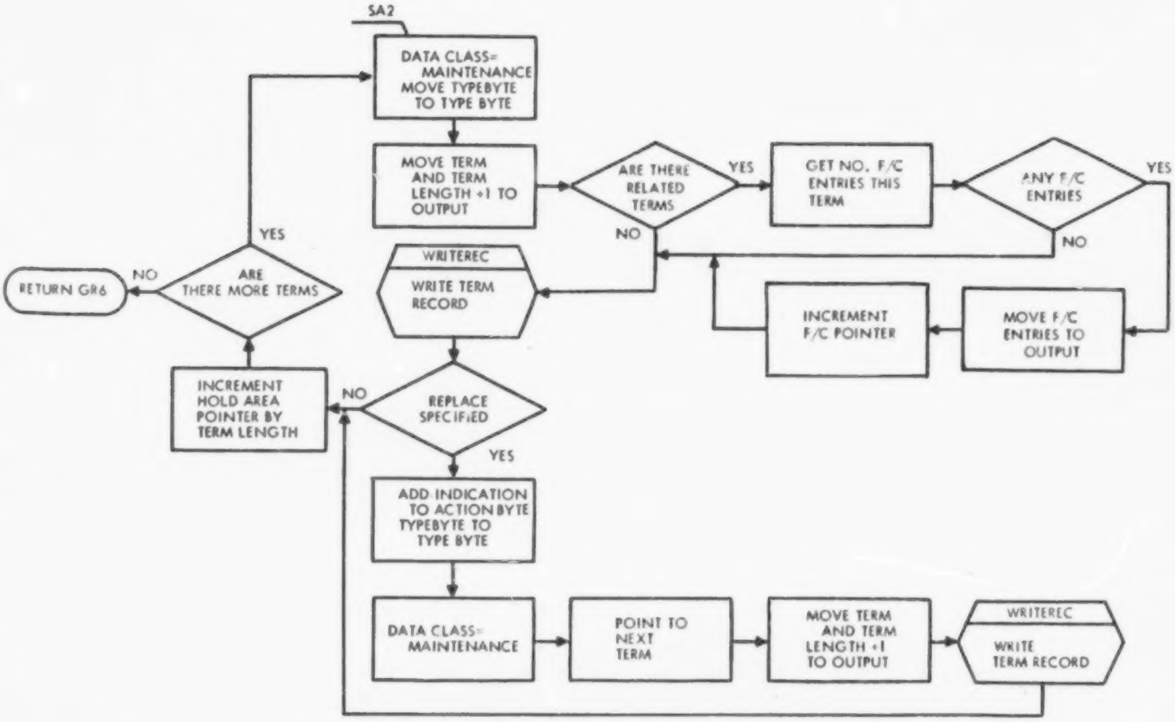

PROCESSING REPLACE
YES
ERROR 99
LOOP
NO
FILE/CATEGORIES FOUND BEFORE
YES
FC FOR RELATED TERM
NO
CAN IT BE MAIN TERM FC
YES
INDICATE IN OPTION SWITCH THAT FC FOUND
NO
ERROR 56
YES
IS THIS A DELETE DATA SET
NO
YES
ERROR 69
NO
INDICATE FC/ FOUND-INITIALIZE FC/POINTER
POINT TO NEXT FC HOLD POSITION
FILE AND CATEGORY SEPARATED BY "-"
NO
ERROR 50
LOOP2
YES
POINT TO FILE TABLE
FILECK
CHECK FILE MNEMONIC FOR VALIDITY
STORE BINARY FILE NUMBER IN HOLD AREA
SPECIAL CHARACTER AFTER CATEGORY X'00' OR
NO
ERROR 51
YES
IS CORRECT CATEGORY TABLE IN CORE
YES
NO
LOAD APPROPRIATE FILE
POINT TO CATEGORY TABLE
FILECK
CHECK CATEGORY MNEMONIC FOR VALIDITY
MOVE BINARY CATEGORY NUMBER TO HOLD AREA
STORE NEW HOLD AREA POINTER
INCREMENT F/C COUNTER
> 20 THIS TERM
YES
ERROR 49
AEND
NO

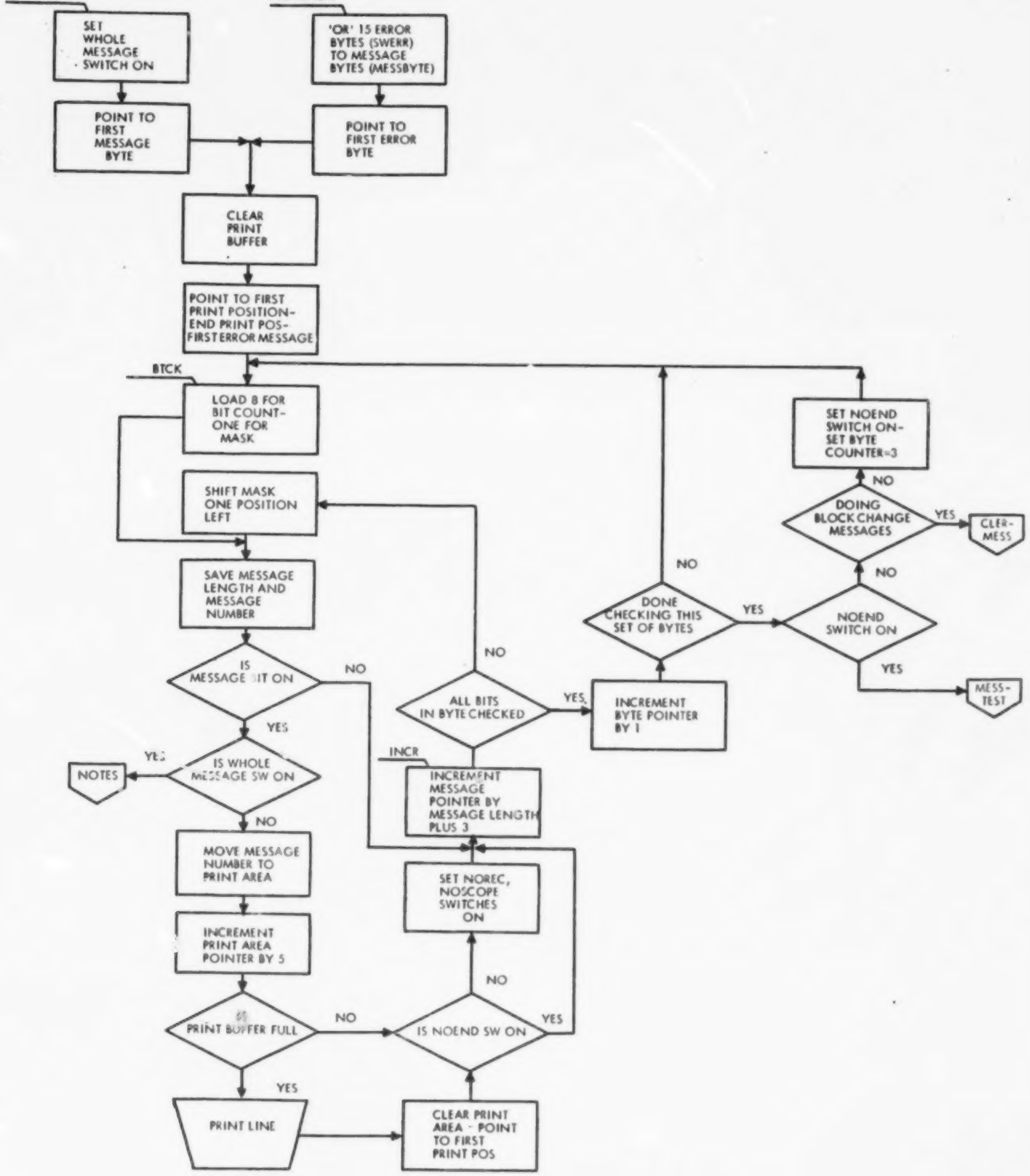
DEDIT

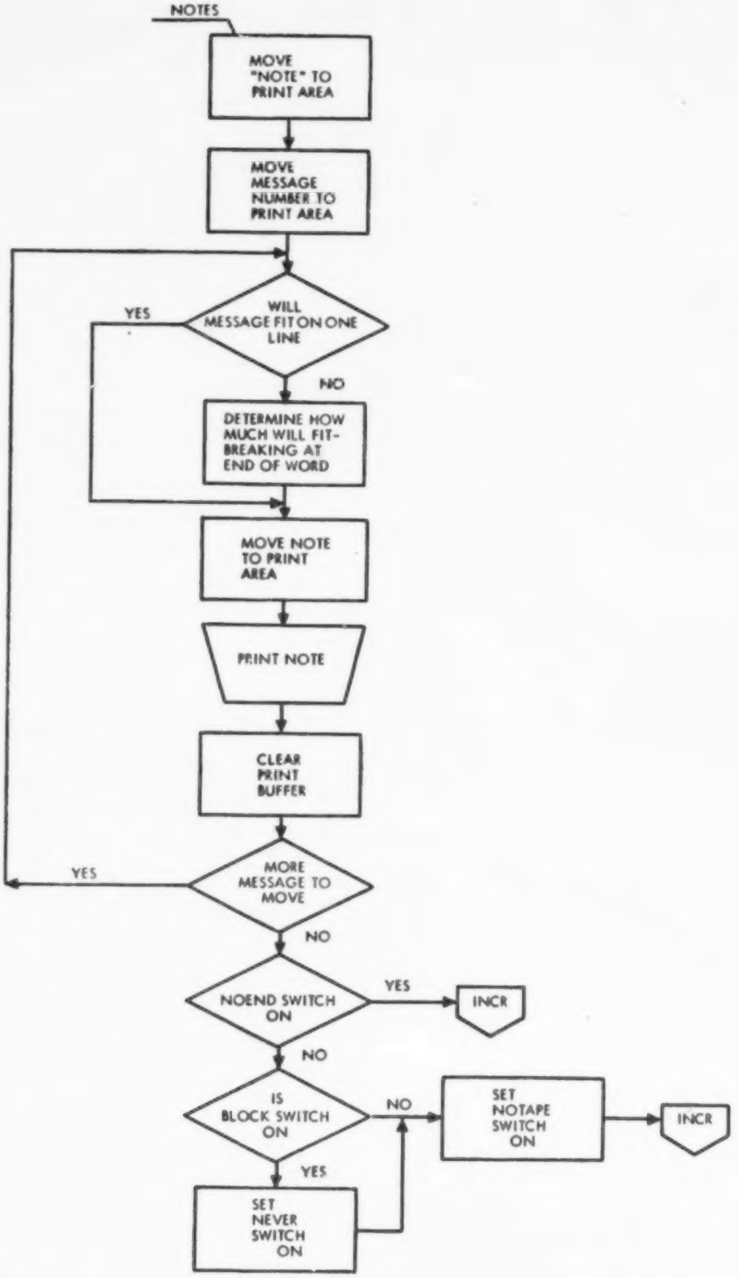

FORMREC
DELETE "ALL" RECORD
YES
ACTION=DELETE DATA CLASS= TERM MAINT TYPE=MAIN TERM
WRITEREC
WRITE RECORD
RETURN GR7
CLEAR 79 POSITION OUTPUT AREA
FUNCTION GIVEN
NO
YES
FUNCTION AND ACTION SWITCHES TO OUTPUT BUFFER
IS THIS A NEW TERM NAME
YES
DATA CLASS= NEW TERM NEW TERM NAME TO OUTPUT
WRITEREC
WRITE NEW TERM RECORD
NO
DATA CLASS= FUNCTION
WRITEREC
WRITE FUNCTION RECORD
ARE THERE DICTIONARY ENTRIES
NO
YES
DATA CLASS= DICTIONARY MOVE DICTIONARY NUMBERS TO OUTPUT BUFFER
WRITEREC
WRITE DICTIONARY RECORD
SCOPE CK
WRITEREC
WRITE FC/ RECORD
DATA CLASS=FC GET FC NO. FOR MAIN TERM, MOVE F/C ENTRIES TO OUTPUT AREA
YES
CHECKFC
ARE THERE MAIN TERM F/C ENTRIES
NO
CHECK FC
WRITEREC
WRITE AUTHORITY RECORD
CHECKSA
DATA CLASS= AUTHORITY AUTHORITY NAME TO OUTPUT AREA
YES
AUTHORITY GIVEN
NO
CHECKSA
SA2
BUILD USE RECORD
POINT TO USE HOLD AREA TYPE BYTE=X'01'
YES
IS THERE A USE TERM
NO
ENTREP
ACTION BYTE TO OUTPUT

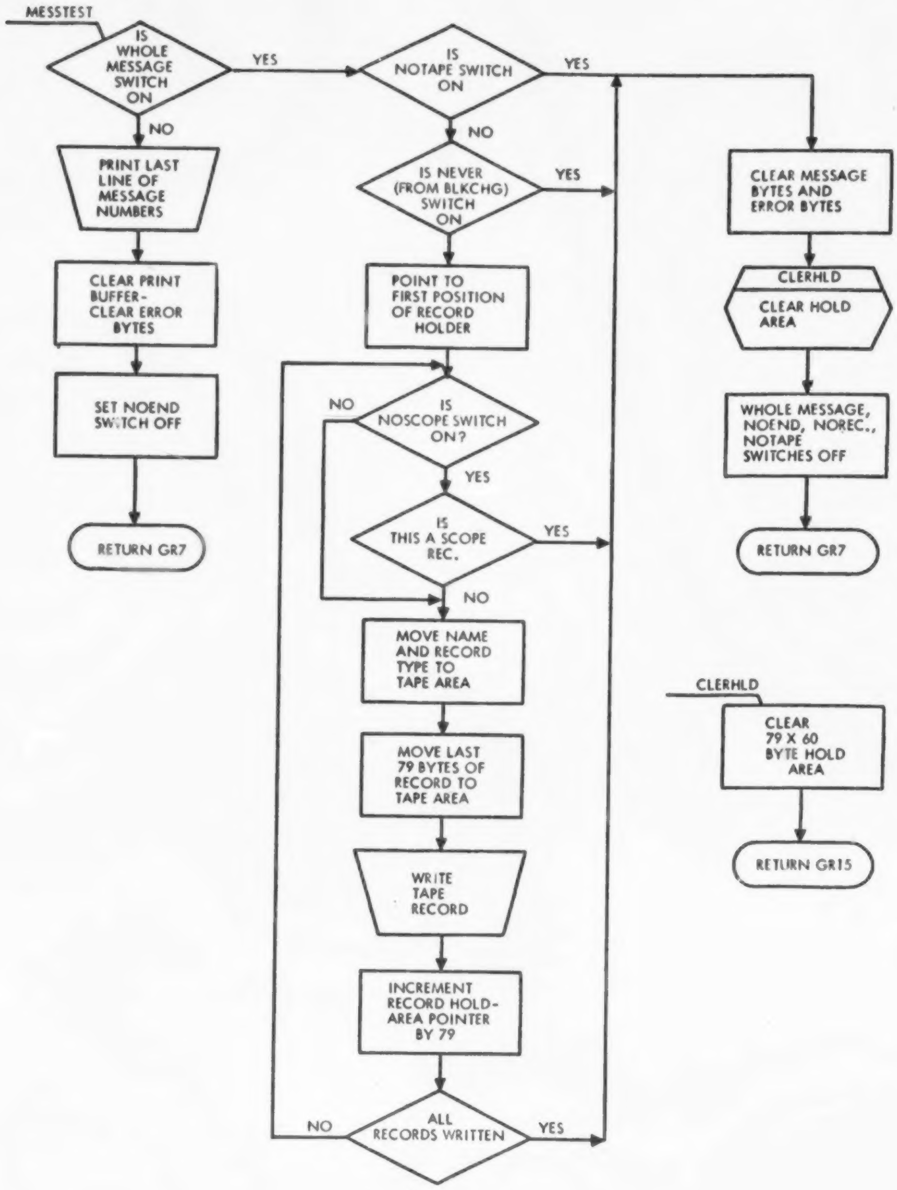

SCOPECK
DELETE SCOPE SPECIFIED
NO
CHECK FC
YES
'ALL' SPECIFIED
YES
'ALL', TYPE AND DATA CLASS CODES TO OUTPUT
WRITEREC
PUT OUTPUT RECORD
CHECK FC
NO
GET HOW MANY TO BE DELETED; MAXIMUM MOVE =64; POINT TO NUMBERS
DATA CLASS AND TYPE CODES TO OUTPUT
MORE THAN 64 NOTE NUMBERS
NO
MOVE REMAINING NUMBERS AND LENGTH TO OUTPUT
WRITEREC
PUT OUTPUT RECORD
CHECK FC
YES
MOVE 64 NOTE NUMBER AND A LENGTH TO OUTPUT RECORD
SUBTRACT 64 FROM NUMBER OF NUMBERS, POINT TO NEXT NUMBERS
WRITEREC
PUT OUTPUT RECORD

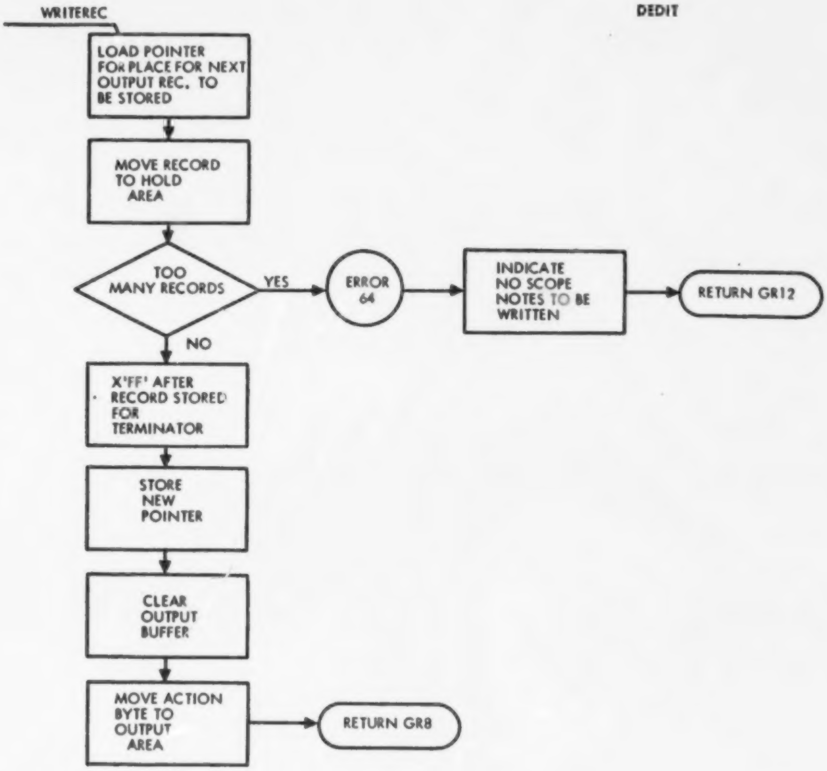

CHECKSA
NO
SEE ALSO TERMS GIVEN
YES
POINT TO SEE ALSO HOLD AREA - TYPEBYTE=X'04'
SA2
MAKE SEE ALSO RECORDS
RELATED TERMS GIVEN
NO
RETURN GR7
YES
POINT TO RELATED TERM HOLD AREA +1 TYPEBYTE=X'02'
SA2
MAKE RELATED TERM RECORDS

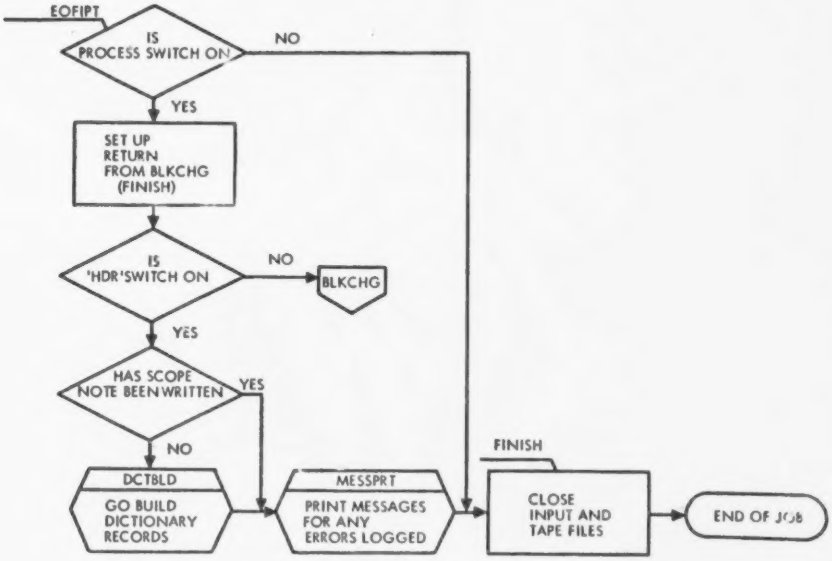

SA2
DATA CLASS= MAINTENANCE MOVE TYPEBYTE TO TYPE BYTE
MOVE TERM AND TERM LENGTH +1 TO OUTPUT
ARE THERE RELATED TERMS
YES
GET NO. F/C ENTRIES THIS TERM
ANY F/C ENTRIES
YES
NO
NO
WRITEREC
WRITE TERM RECORD
INCREMENT F/C POINTER
MOVE F/C ENTRIES TO OUTPUT
YES
RETURN GR6
NO
ARE THERE MORE TERMS
INCREMENT HOLD AREA POINTER BY TERM LENGTH
NO
REPLACE SPECIFIED
YES
ADD INDICATION TO ACTIONBYTE TYPEBYTE TO TYPE BYTE
DATA CLASS= MAINTENANCE
POINT TO NEXT TERM
MOVE TERM AND TERM LENGTH +1 TO OUTPUT
WRITEREC
WRITE TERM RECORD


MESSPRT
SET WHOLE MESSAGE SWITCH ON
POINT TO FIRST MESSAGE BYTE
NOTENUM
'OR' 15 ERROR BYTES (SWERR) TO MESSAGE BYTES (MESSBYTE)
POINT TO FIRST ERROR BYTE
CLEAR PRINT BUFFER
POINT TO FIRST PRINT POSITION-END PRINT POS-FIRST ERROR MESSAGE
BTCK
LOAD 8 FOR BIT COUNT-ONE FOR MASK
SHIFT MASK ONE POSITION LEFT
SAVE MESSAGE LENGTH AND MESSAGE NUMBER
IS MESSAGE BIT ON
NO
YES
IS WHOLE MESSAGE SW ON
YES
NOTES
NO
MOVE MESSAGE NUMBER TO PRINT AREA
INCREMENT PRINT AREA POINTER BY 5
IS PRINT BUFFER FULL
NO
YES
PRINT LINE
CLEAR PRINT AREA - POINT TO FIRST PRINT POS
IS NOEND SW ON
NO
YES
SET NOREC, NOSCOPE SWITCHES ON
INCR
INCREMENT MESSAGE POINTER BY MESSAGE LENGTH PLUS 3
ALL BITS IN BYTE CHECKED
NO
YES
INCREMENT BYTE POINTER BY 1
DONE CHECKING THIS SET OF BYTES
NO
YES
NOEND SWITCH ON
YES
MESS-TEST
NO
DOING BLOCK CHANGE MESSAGES
YES
CLER-MESS
NO
SET NOEND SWITCH ON-SET BYTE COUNTER=3


NOTES
MOVE "NOTE" TO PRINT AREA
MOVE MESSAGE NUMBER TO PRINT AREA
WILL MESSAGE FIT ON ONE LINE
YES
NO
DETERMINE HOW MUCH WILL FIT-BREAKING AT END OF WORD
MOVE NOTE TO PRINT AREA
PRINT NOTE
CLEAR PRINT BUFFER
MORE MESSAGE TO MOVE
YES
NO
NOEND SWITCH ON
YES
INCR
NO
IS BLOCK SWITCH ON
NO
SET NOTAPE SWITCH ON
INCR
YES
SET NEVER SWITCH ON


MESSTEST
IS WHOLE MESSAGE SWITCH ON
YES
NO
PRINT LAST LINE OF MESSAGE NUMBERS
CLEAR PRINT BUFFER- CLEAR ERROR BYTES
SET NOEND SWITCH OFF
RETURN GR7
IS NOTAPE SWITCH ON
YES
NO
IS NEVER (FROM BLKCHG) SWITCH ON
YES
POINT TO FIRST POSITION OF RECORD HOLDER
NO
IS NOSCOPE SWITCH ON?
YES
IS THIS A SCOPE REC.
YES
NO
MOVE NAME AND RECORD TYPE TO TAPE AREA
MOVE LAST 79 BYTES OF RECORD TO TAPE AREA
WRITE TAPE RECORD
INCREMENT RECORD HOLD-AREA POINTER BY 79
NO
ALL RECORDS WRITTEN
YES
CLEAR MESSAGE BYTES AND ERROR BYTES
CLERHLD
CLEAR HOLD AREA
WHOLE MESSAGE, NOEND, NOREC., NOTAPE SWITCHES OFF
RETURN GR7
CLERHLD
CLEAR 79 X 60 BYTE HOLD AREA
RETURN GR15


WRITEREC
LOAD POINTER FOR PLACE FOR NEXT OUTPUT REC. TO BE STORED
MOVE RECORD TO HOLD AREA
TOO MANY RECORDS
YES
ERROR 64
INDICATE NO SCOPE NOTES TO BE WRITTEN
RETURN GR12
NO
X'FF' AFTER RECORD STORED FOR TERMINATOR
STORE NEW POINTER
CLEAR OUTPUT BUFFER
MOVE ACTION BYTE TO OUTPUT AREA
RETURN GR8


EOFIPT
IS PROCESS SWITCH ON
NO
YES
SET UP RETURN FROM BLKCHG (FINISH)
IS 'HDR'SWITCH ON
NO
BLKCHG
YES
HAS SCOPE NOTE BEEN WRITTEN
YES
NO
DCTBLD
GO BUILD DICTIONARY RECORDS
MESSPRT
PRINT MESSAGES FOR ANY ERRORS LOGGED
FINISH
CLOSE INPUT AND TAPE FILES
END OF JOB

DICTIONARY MAINTENANCE PROGRAM (DICT2) PROCESSING LOGIC

During the initialization procedure, all four tape files are opened, switches and counters are set, the index tables are readied for disk output and the first records on each input file are read. If there are any block changes, these are read and stored in the block change table.

1. Compare

   Compare maintenance update to prior dictionary record.

2. Update Low

   If the update is low, create a dictionary record for this new term and read and process each additional update record for this term. Output the new dictionary record and return to the Compare above.

3. Update High

   Scan the dictionary record for any block changes and make any changes required. Output this dictionary record with block change, if any. Read the next dictionary input record and return to the Compare above.

4. Update Equal

   If "delete all", output this dictionary record on the change/error tape, read the next update record and return to the Compare above. If not "delete all", scan for any block changes and change any matching elements as requested. Read and process each update record for this term. Output this dictionary record with the changes made.

Read the next dictionary record and return to the Compare above.

5. End of File

After either end of file, pad that file with high order values and process the remainder of the other file. After both end of files on the input tapes, close all four tape files, finish writing the index tables on disk, and return to the supervisor program.

6. Scan for Block Changes

Scan the dictionary record once for each block change record. If there is a match, that term is either deleted from the dictionary record or replaced by the replacing term. If there are no block changes, the dictionary records are not scanned. If there is no match for any of the block changes, the dictionary record remains unchanged.

7. Process Update Record Routine

Before changing any dictionary record, the input (from the beginning through the file/context codes) is moved to an output area and the remainder of the record, starting with the main term, is moved to a temporary output area. All changes are made in these output areas and the input area is left unchanged. This update routine will add or delete a term, a scope note, a dictionary code, a file/context code, or a function. To add or delete a term, the dictionary output area is scanned for that term. If it is an "add", and a hit is found, or if it is a "delete" and no hit is found, an error switch is set. Otherwise, for a delete the record is flagged as such, and for an addition, the new term is inserted at the end of the output area. Scope notes, dictionaries, and file/context codes are handled in the same manner.

After all changes have been made, the temporary output area is sorted into the output area immediately following the file/context codes. The elements are sorted by term type, then alphabetically by term. Those terms with delete flags (term type X'FO')are discarded.

8. Output Routine

If there are any changes to the dictionary record or any errors, one or more message notes are written on the change/error tape, followed by the prior and/or new dictionary records. If the request is a "delete" with no errors, control returns to the main program. If there was no change to the original record, or if there were any errors in the update information, the unchanged original dictionary record is written on tape and disk. If there were no errors, the changed dictionary record is output. Control then returns to the main program.

9. DADIO

The disk output routine blocks the condensed dictionary records and writes this data on the disk one full track at a time. (3625 characters per track) As each track is completed and written on disk, a track index is updated with the main term for the last record on that track. For each ten tracks, a cylinder index is updated and for each 50 cylinders the master index is updated.

Input

DICT2 requires two tapes as input.

1. The sorted tape of maintenance changes produced by DEDIT will be mounted on SYS006. Its format is shown below labeled D1. The records are 185 bytes long blocked by a factor of 10 into 1850 byte blocks.

2. The prior dictionary tape is mounted on SYS007. Its positional format is labeled D2 below. In addition a complete layout appears in APPENDIX A. Its format is variable in length and blocked.

   In addition to the tapes DICT2 allows a UPSI card which will prevent the disk from being written. The format is //UPSI 10000000.

Output

DICT2 produces two tapes and a disk file.

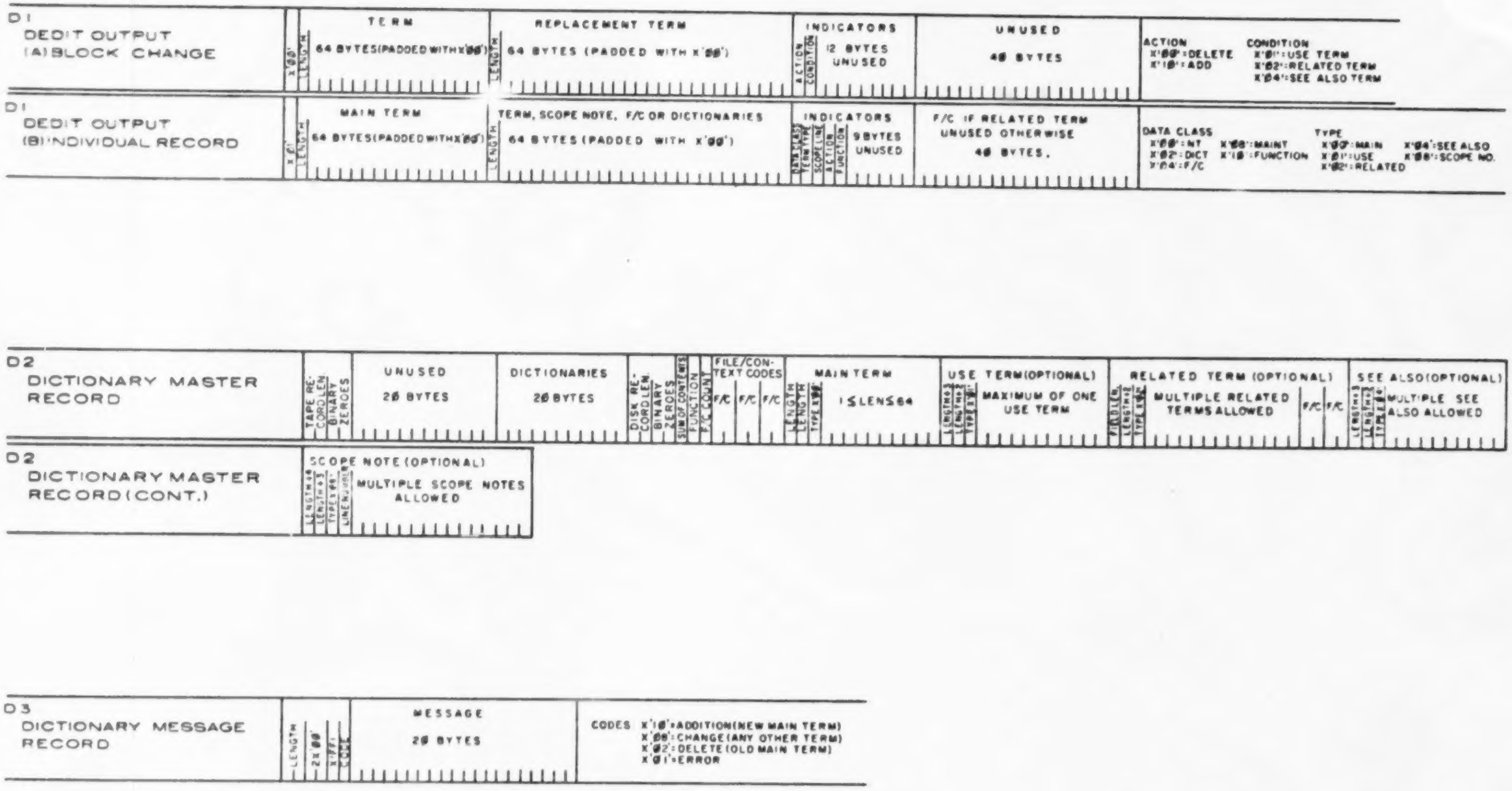
1. Tape

   The tape of alterations performed (message tape) will be written on SYS008. It will contain two types of records: Messages (labeled D3 below) and Dictionary records (labeled D2 below).

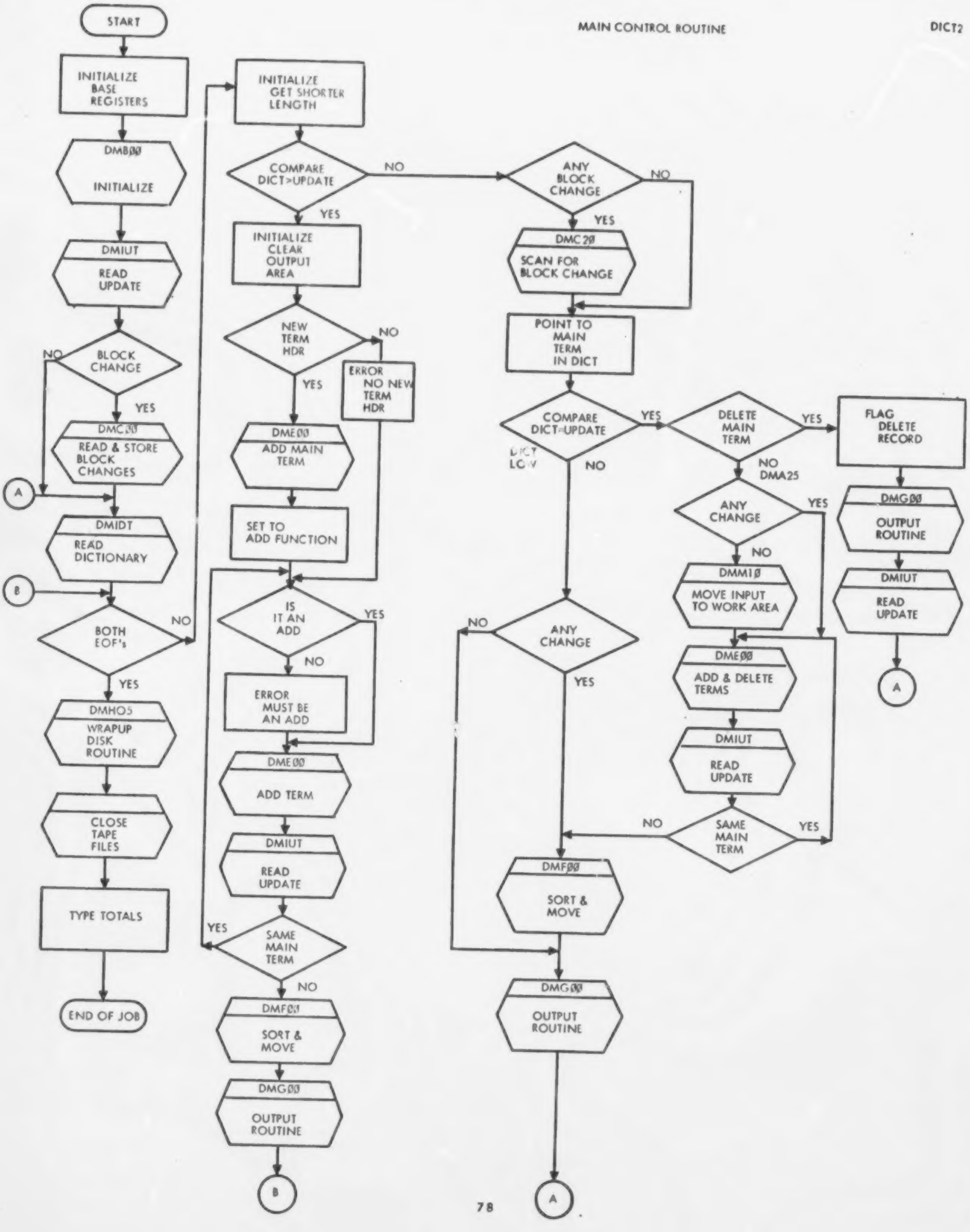
   The updated dictionary tape will be in exactly the same format as the prior dictionary tape.

2. Disk
   If created the disk record format will be a subset of the tape record format. The first 44 bytes are not included. The record begins with the field labeled 'DISK RECORD LENGTH' and concludes with the 'RELATED TERM' field.

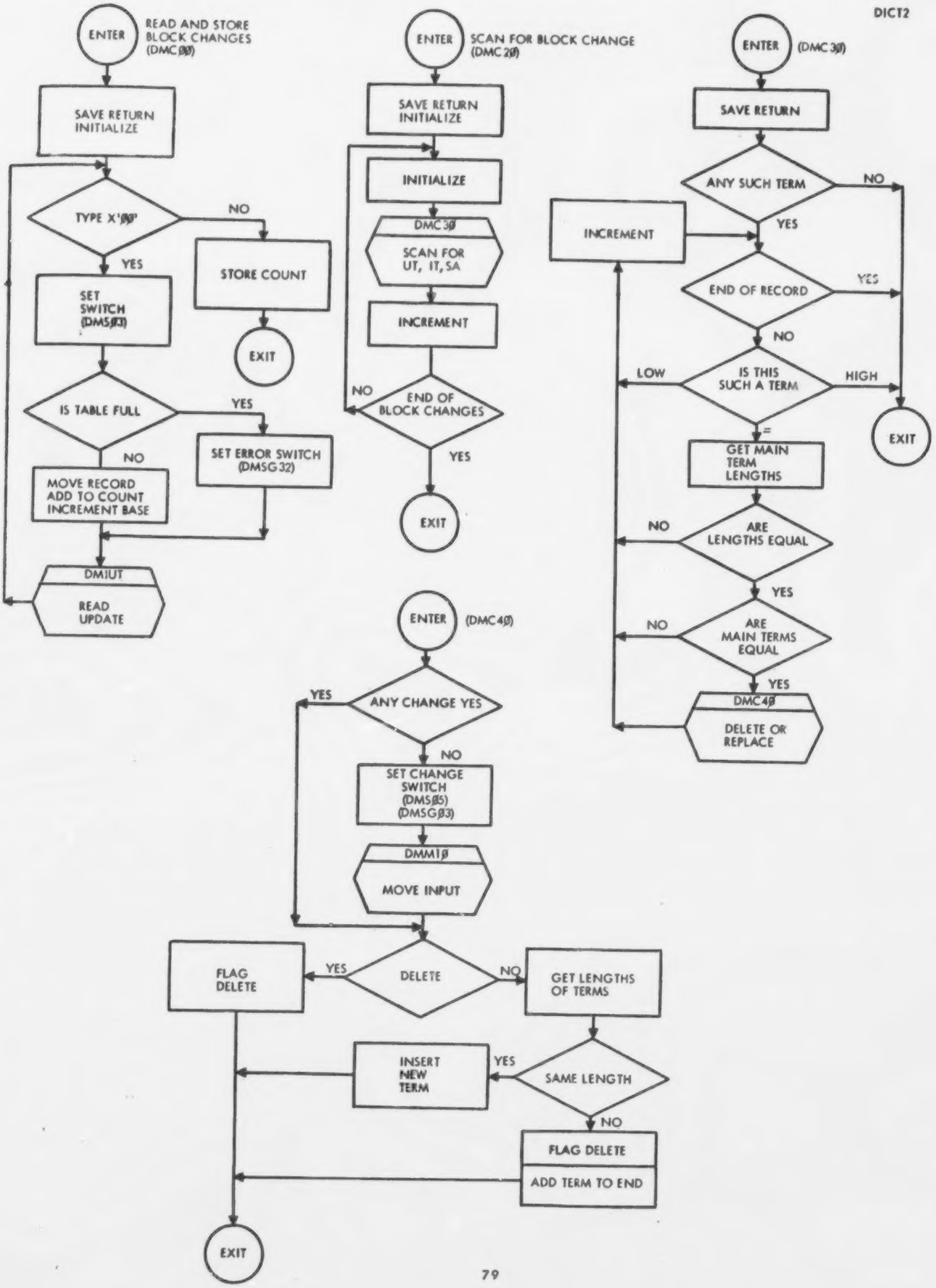
DICTIONARY MAINTENANCE PROGRAM (DICT2) RECORD FORMATS


D1
DEDIT OUTPUT
(A) BLOCK CHANGE
X'00'
LENGTH
TERM
64 BYTES (PADDED WITH X'00')
LENGTH
REPLACEMENT TERM
64 BYTES (PADDED WITH X'00')
INDICATORS
ACTION
CONDITION
12 BYTES UNUSED
UNUSED
40 BYTES
ACTION
X'00':DELETE
X'10':ADD
CONDITION
X'01':USE TERM
X'02':RELATED TERM
X'04':SEE ALSO TERM
D1
DEDIT OUTPUT
(B) INDIVIDUAL RECORD
X'01'
LENGTH
MAIN TERM
64 BYTES (PADDED WITH X'00')
LENGTH
TERM, SCOPE NOTE, F/C OR DICTIONARIES
64 BYTES (PADDED WITH X'00')
INDICATORS
DATA CLASS
TERM TYPE
SCOPE LINE
ACTION
FUNCTION
9 BYTES UNUSED
F/C IF RELATED TERM
UNUSED OTHERWISE
40 BYTES.
DATA CLASS
X'00':NT
X'02':DICT
X'04':F/C
X'08':MAINT
X'10':FUNCTION
TYPE
X'00':MAIN
X'01':USE
X'02':RELATED
X'04':SEE ALSO
X'08':SCOPE NO.
D2
DICTIONARY MASTER RECORD
TAPE RECORD LEN.
BINARY ZEROES
UNUSED
20 BYTES
DICTIONARIES
20 BYTES
DISK RECORD LEN.
BINARY ZEROES
SUM OF CONTENTS
FUNCTION
F/C COUNT
FILE/CONTEXT CODES
F/C
F/C
F/C
MAIN TERM
LENGTH
LENGTH
TYPE X'00'
1≤LEN≤64
USE TERM (OPTIONAL)
MAXIMUM OF ONE USE TERM
RELATED TERM (OPTIONAL)
MULTIPLE RELATED TERMS ALLOWED
F/C
F/C
SEE ALSO (OPTIONAL)
MULTIPLE SEE ALSO ALLOWED
D2
DICTIONARY MASTER RECORD (CONT.)
SCOPE NOTE (OPTIONAL)
LINE NUMBER
MULTIPLE SCOPE NOTES ALLOWED
D3
DICTIONARY MESSAGE RECORD
LENGTH
2X'00'
X'FF'
CODE
MESSAGE
20 BYTES
CODES
X'10':ADDITION (NEW MAIN TERM)
X'08':CHANGE (ANY OTHER TERM)
X'02':DELETE (OLD MAIN TERM)
X'01':ERROR

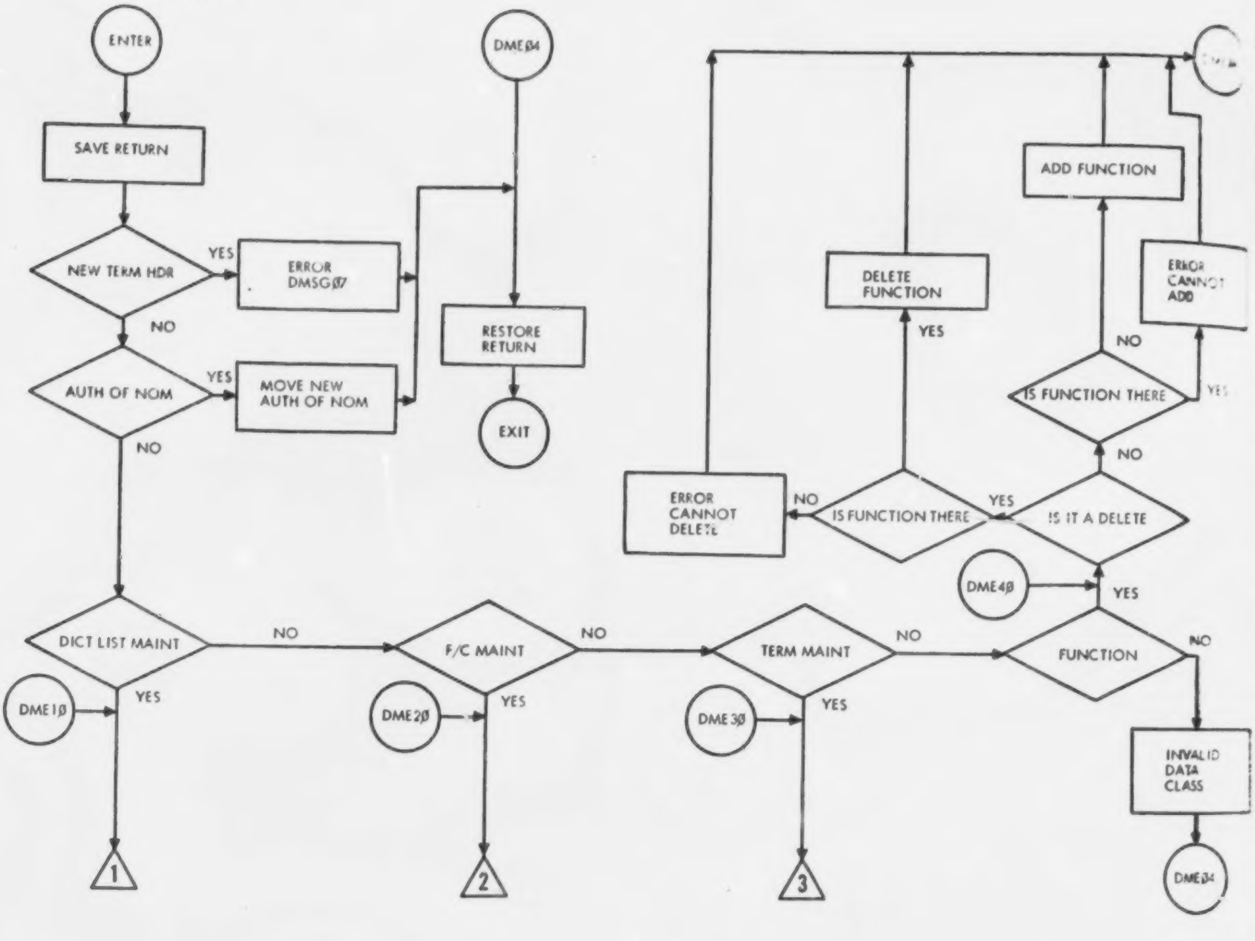
MAIN CONTROL ROUTINE DICT2


START
INITIALIZE BASE REGISTERS
DMB00
INITIALIZE
DMIUT
READ UPDATE
BLOCK CHANGE
NO
YES
DMC00
READ & STORE BLOCK CHANGES
A
DMIDT
READ DICTIONARY
B
BOTH EOF's
NO
YES
DMHO5
WRAPUP DISK ROUTINE
CLOSE TAPE FILES
TYPE TOTALS
END OF JOB
INITIALIZE GET SHORTER LENGTH
COMPARE DICT>UPDATE
NO
YES
INITIALIZE CLEAR OUTPUT AREA
NEW TERM HDR
NO
YES
ERROR NO NEW TERM HDR
DME00
ADD MAIN TERM
SET TO ADD FUNCTION
IS IT AN ADD
YES
NO
ERROR MUST BE AN ADD
DME00
ADD TERM
DMIUT
READ UPDATE
SAME MAIN TERM
YES
NO
DMF00
SORT & MOVE
DMG00
OUTPUT ROUTINE
B
ANY BLOCK CHANGE
NO
YES
DMC20
SCAN FOR BLOCK CHANGE
POINT TO MAIN TERM IN DICT
COMPARE DICT=UPDATE
YES
DICT LOW
NO
ANY CHANGE
NO
YES
DMF00
SORT & MOVE
DMG00
OUTPUT ROUTINE
A
DELETE MAIN TERM
YES
NO
DMA25
ANY CHANGE
YES
NO
DMM10
MOVE INPUT TO WORK AREA
DME00
ADD & DELETE TERMS
DMIUT
READ UPDATE
SAME MAIN TERM
NO
YES
FLAG DELETE RECORD
DMG00
OUTPUT ROUTINE
DMIUT
READ UPDATE
A

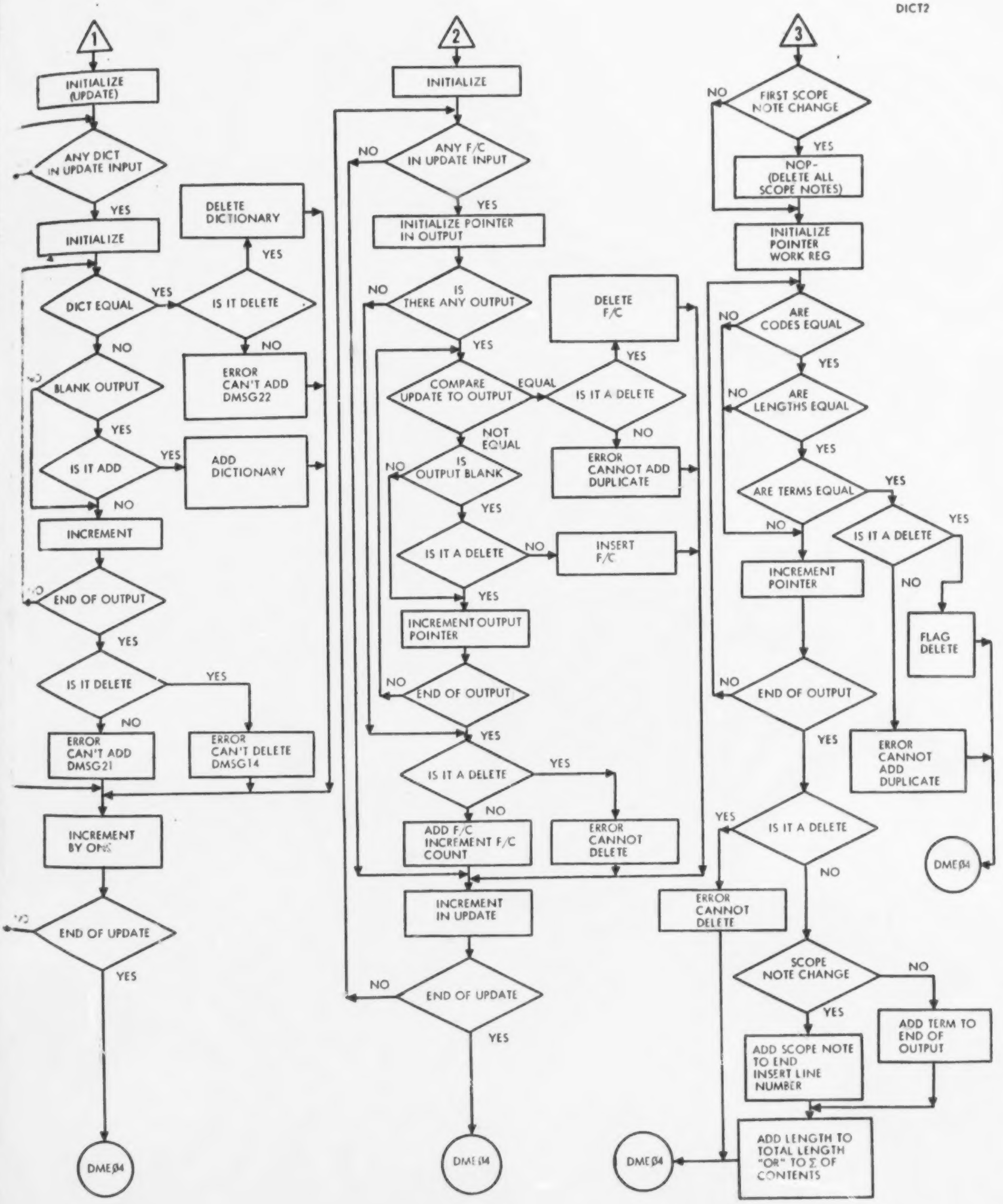
DICT2

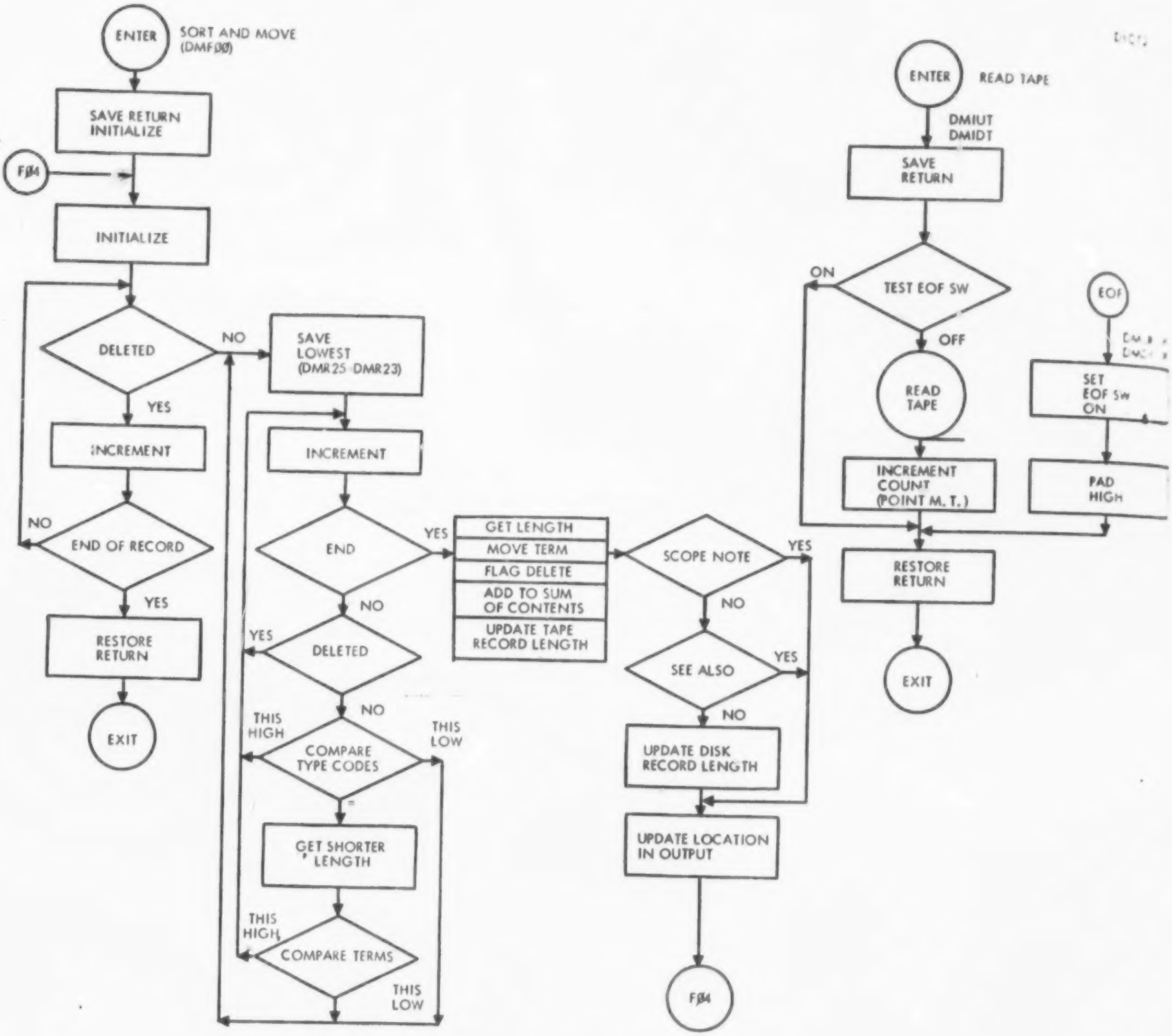

ENTER
READ AND STORE BLOCK CHANGES (DMC00)
SAVE RETURN INITIALIZE
TYPE X'00'
NO
YES
STORE COUNT
EXIT
SET SWITCH (DMS03)
IS TABLE FULL
YES
NO
SET ERROR SWITCH (DMSG32)
MOVE RECORD ADD TO COUNT INCREMENT BASE
DMIUT
READ UPDATE
ENTER
SCAN FOR BLOCK CHANGE (DMC20)
SAVE RETURN INITIALIZE
INITIALIZE
DMC30
SCAN FOR UT, IT, SA
INCREMENT
END OF BLOCK CHANGES
NO
YES
EXIT
ENTER
(DMC30)
SAVE RETURN
ANY SUCH TERM
NO
YES
INCREMENT
END OF RECORD
YES
NO
IS THIS SUCH A TERM
LOW
HIGH
=
EXIT
GET MAIN TERM LENGTHS
ARE LENGTHS EQUAL
NO
YES
ARE MAIN TERMS EQUAL
NO
YES
DMC40
DELETE OR REPLACE
ENTER
(DMC40)
ANY CHANGE YES
YES
NO
SET CHANGE SWITCH (DMS05) (DMSG03)
DMM10
MOVE INPUT
DELETE
YES
NO
FLAG DELETE
GET LENGTHS OF TERMS
SAME LENGTH
YES
NO
INSERT NEW TERM
FLAG DELETE
ADD TERM TO END
EXIT

ADD OR DELETE
(SCAN TEMP OUTPUT)
(DME00)

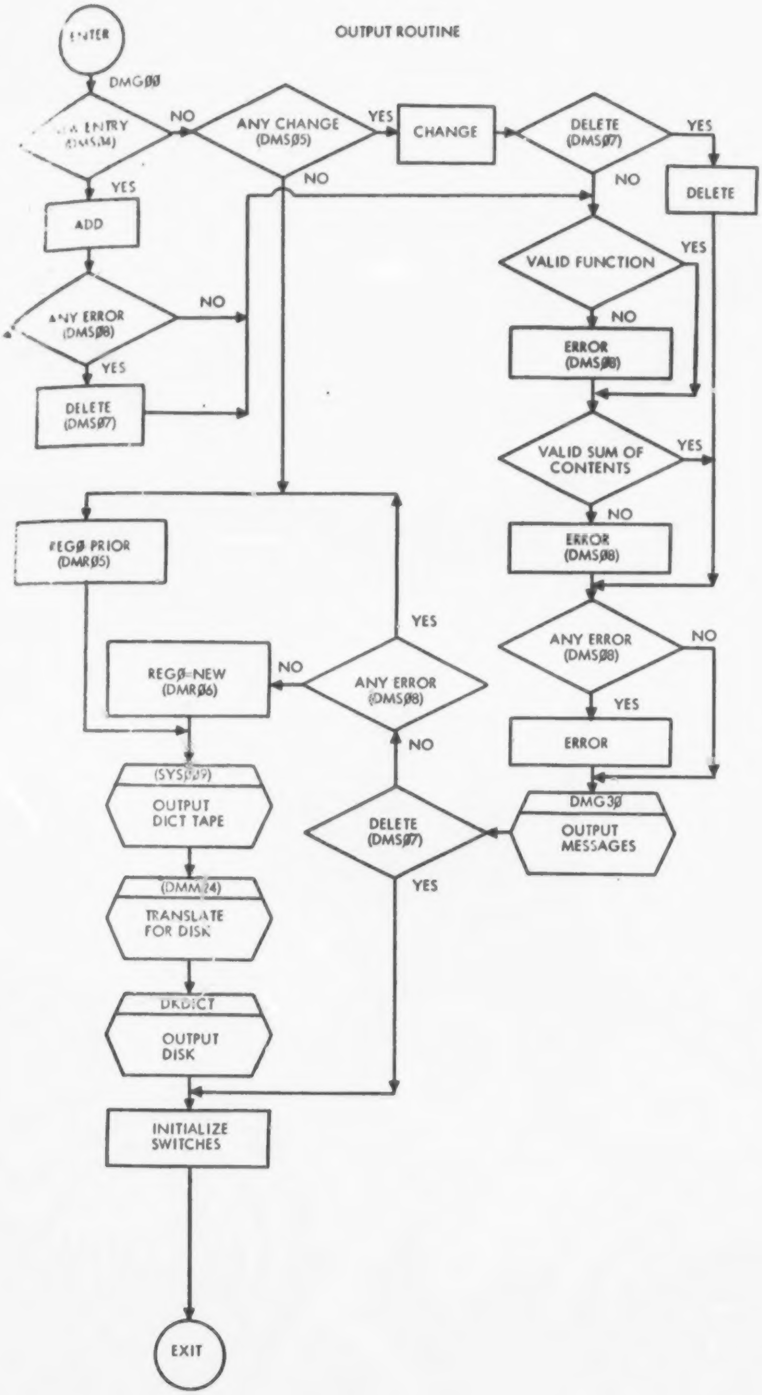

ENTER
SAVE RETURN
NEW TERM HDR
YES
ERROR DMSG07
NO
AUTH OF NOM
YES
MOVE NEW AUTH OF NOM
NO
DME04
RESTORE RETURN
EXIT
DICT LIST MAINT
NO
YES
DME10
1
F/C MAINT
NO
YES
DME20
2
TERM MAINT
NO
YES
DME30
3
FUNCTION
NO
YES
INVALID DATA CLASS
DME04
DME40
IS IT A DELETE
YES
NO
IS FUNCTION THERE
NO
YES
ERROR CANNOT DELETE
DELETE FUNCTION
IS FUNCTION THERE
NO
YES
ADD FUNCTION
ERROR CANNOT ADD

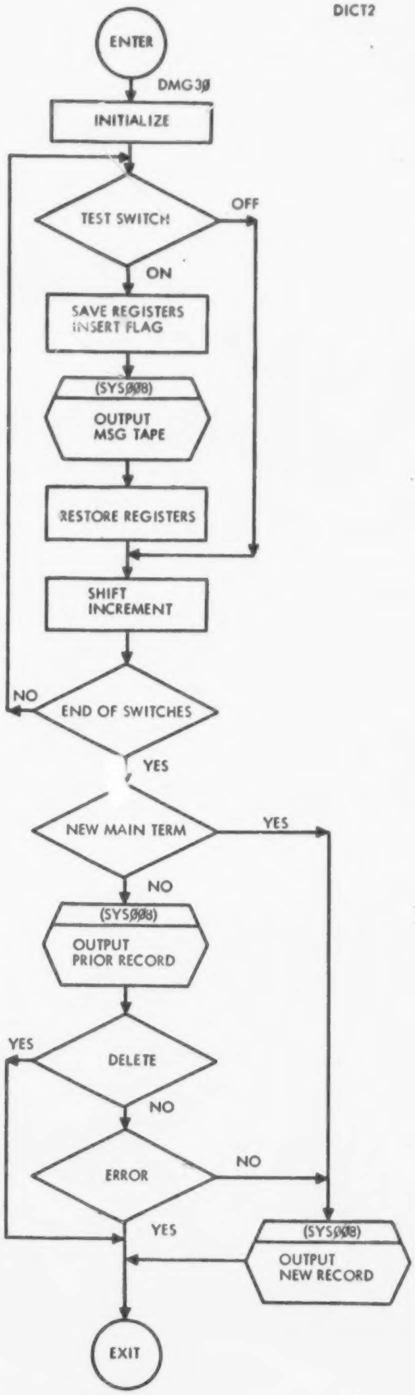
DICT2

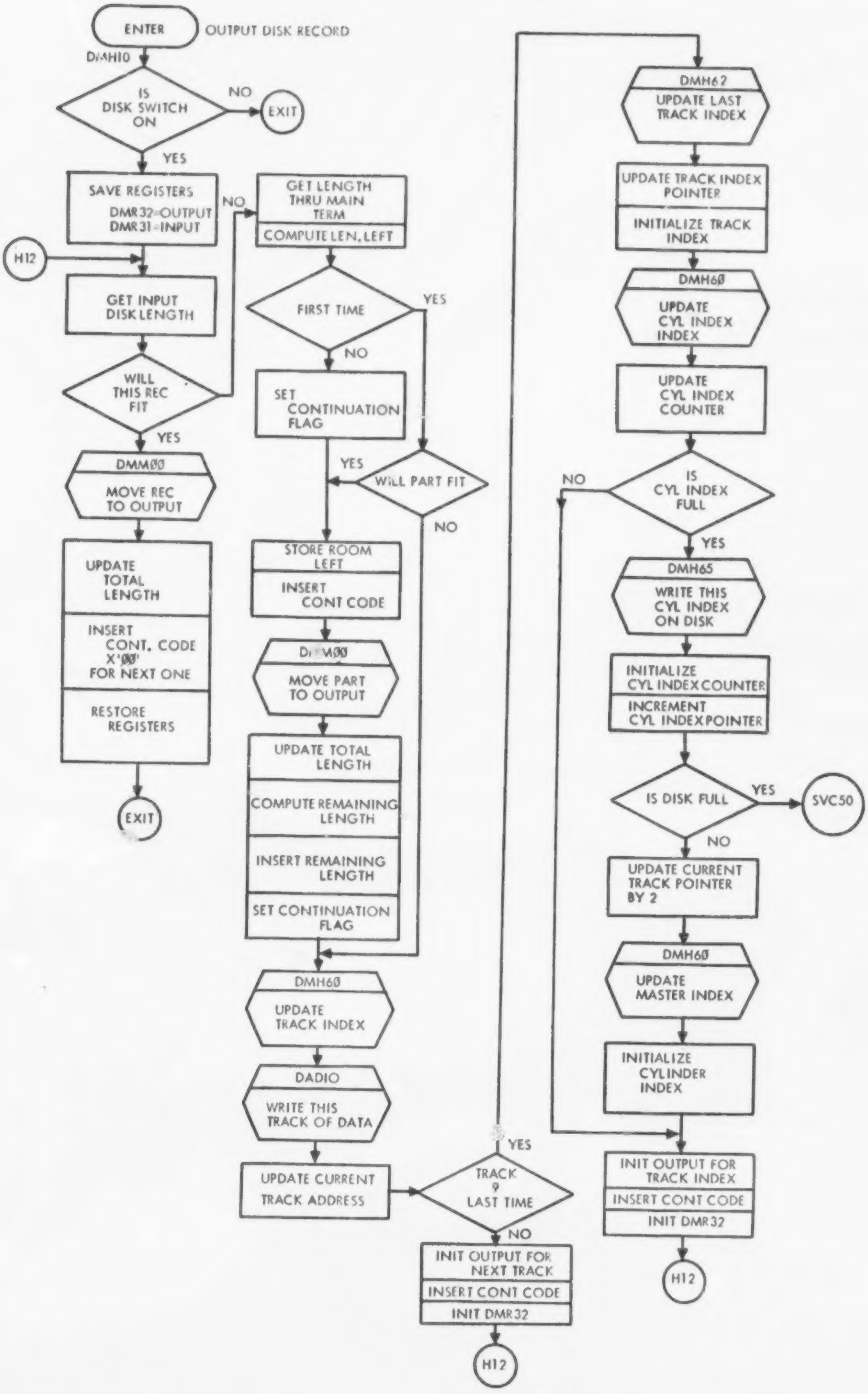

1
INITIALIZE (UPDATE)
ANY DICT IN UPDATE INPUT
YES
INITIALIZE
DICT EQUAL
YES
IS IT DELETE
YES
DELETE DICTIONARY
NO
ERROR CAN'T ADD DMSG22
NO
BLANK OUTPUT
YES
IS IT ADD
YES
ADD DICTIONARY
NO
INCREMENT
END OF OUTPUT
YES
IS IT DELETE
YES
ERROR CAN'T DELETE DMSG14
NO
ERROR CAN'T ADD DMSG21
INCREMENT BY ONE
END OF UPDATE
YES
DMEØ4
2
INITIALIZE
ANY F/C IN UPDATE INPUT
NO
YES
INITIALIZE POINTER IN OUTPUT
IS THERE ANY OUTPUT
NO
YES
COMPARE UPDATE TO OUTPUT
EQUAL
IS IT A DELETE
YES
DELETE F/C
NO
ERROR CANNOT ADD DUPLICATE
NOT EQUAL
IS OUTPUT BLANK
NO
YES
IS IT A DELETE
NO
INSERT F/C
YES
INCREMENT OUTPUT POINTER
END OF OUTPUT
NO
YES
IS IT A DELETE
YES
ERROR CANNOT DELETE
NO
ADD F/C INCREMENT F/C COUNT
INCREMENT IN UPDATE
END OF UPDATE
NO
YES
DMEØ4
3
FIRST SCOPE NOTE CHANGE
NO
YES
NOP- (DELETE ALL SCOPE NOTES)
INITIALIZE POINTER WORK REG
ARE CODES EQUAL
NO
YES
ARE LENGTHS EQUAL
NO
YES
ARE TERMS EQUAL
YES
IS IT A DELETE
YES
FLAG DELETE
NO
ERROR CANNOT ADD DUPLICATE
DMEØ4
NO
INCREMENT POINTER
END OF OUTPUT
NO
YES
IS IT A DELETE
YES
ERROR CANNOT DELETE
NO
SCOPE NOTE CHANGE
YES
ADD SCOPE NOTE TO END INSERT LINE NUMBER
NO
ADD TERM TO END OF OUTPUT
ADD LENGTH TO TOTAL LENGTH "OR" TO Σ OF CONTENTS
DMEØ4


ENTER
SORT AND MOVE (DMF00)
SAVE RETURN INITIALIZE
F04
INITIALIZE
DELETED
NO
YES
INCREMENT
NO
END OF RECORD
YES
RESTORE RETURN
EXIT
SAVE LOWEST (DMR25-DMR23)
INCREMENT
END
YES
NO
YES
DELETED
NO
THIS HIGH
COMPARE TYPE CODES
THIS LOW
=
GET SHORTER LENGTH
THIS HIGH
COMPARE TERMS
THIS LOW
GET LENGTH
MOVE TERM
FLAG DELETE
ADD TO SUM OF CONTENTS
UPDATE TAPE RECORD LENGTH
SCOPE NOTE
YES
NO
SEE ALSO
YES
NO
UPDATE DISK RECORD LENGTH
UPDATE LOCATION IN OUTPUT
F04
ENTER
READ TAPE
DMIUT
DMIDT
SAVE RETURN
TEST EOF SW
ON
OFF
READ TAPE
INCREMENT COUNT (POINT M. T.)
RESTORE RETURN
EXIT
EOF
SET EOF SW ON
PAD HIGH

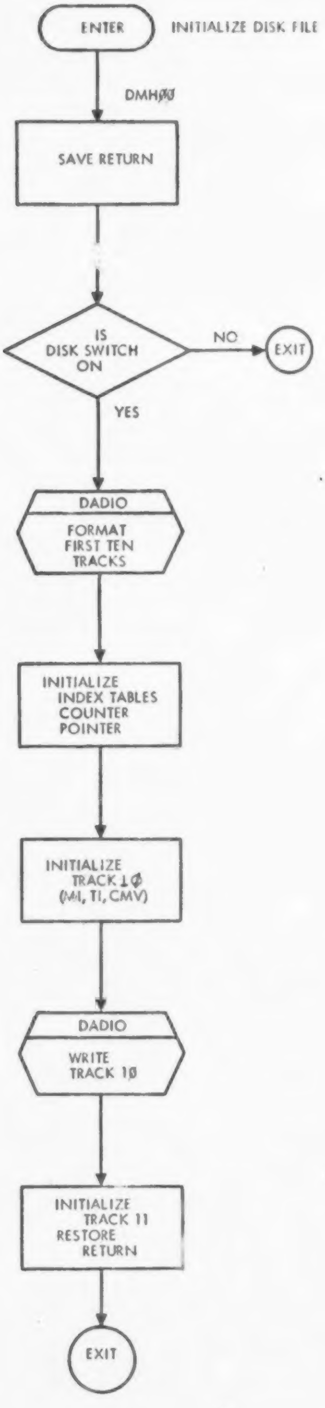
OUTPUT ROUTINE

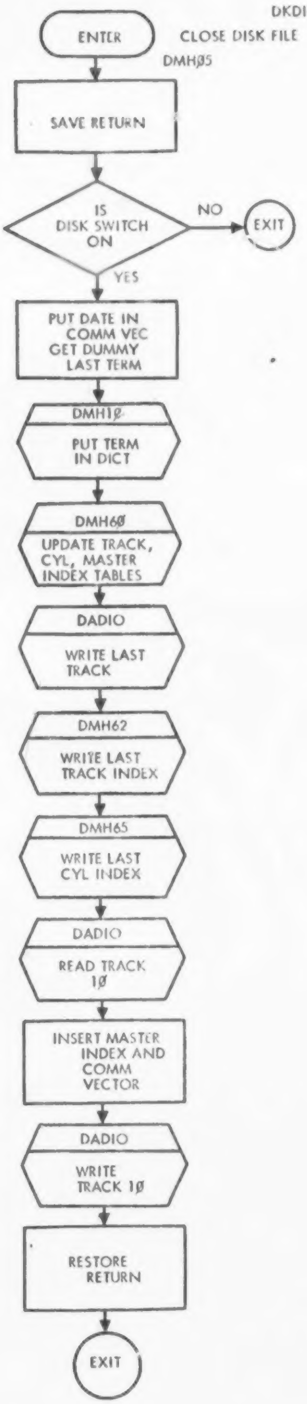

ENTER
DMG00
NEW ENTRY (DMS04)
NO
YES
ADD
ANY ERROR (DMS08)
NO
YES
DELETE (DMS07)
ANY CHANGE (DMS05)
YES
NO
CHANGE
DELETE (DMS07)
YES
NO
DELETE
VALID FUNCTION
YES
NO
ERROR (DMS08)
VALID SUM OF CONTENTS
YES
NO
ERROR (DMS08)
ANY ERROR (DMS08)
NO
YES
ERROR
DMG30
OUTPUT MESSAGES
REG0 PRIOR (DMR05)
REG0=NEW (DMR06)
NO
ANY ERROR (DMS08)
YES
NO
DELETE (DMS07)
YES
(SYS009)
OUTPUT DICT TAPE
(DMM24)
TRANSLATE FOR DISK
DRDICT
OUTPUT DISK
INITIALIZE SWITCHES
EXIT


ENTER
DMG30
INITIALIZE
TEST SWITCH
OFF
ON
SAVE REGISTERS INSERT FLAG
(SYS008)
OUTPUT MSG TAPE
RESTORE REGISTERS
SHIFT INCREMENT
NO
END OF SWITCHES
YES
NEW MAIN TERM
YES
NO
(SYS008)
OUTPUT PRIOR RECORD
YES
DELETE
NO
ERROR
NO
YES
(SYS008)
OUTPUT NEW RECORD
EXIT

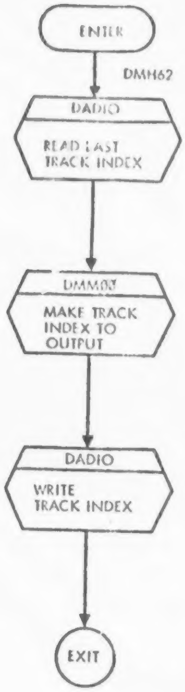

ENTER
OUTPUT DISK RECORD
DMH10
IS DISK SWITCH ON
NO
EXIT
YES
SAVE REGISTERS
DMR32=OUTPUT
DMR31=INPUT
H12
GET INPUT DISK LENGTH
WILL THIS REC FIT
NO
YES
DMM00
MOVE REC TO OUTPUT
UPDATE TOTAL LENGTH
INSERT CONT. CODE X'00' FOR NEXT ONE
RESTORE REGISTERS
EXIT
GET LENGTH THRU MAIN TERM
COMPUTE LEN. LEFT
FIRST TIME
YES
NO
SET CONTINUATION FLAG
YES
WILL PART FIT
NO
STORE ROOM LEFT
INSERT CONT CODE
DMM00
MOVE PART TO OUTPUT
UPDATE TOTAL LENGTH
COMPUTE REMAINING LENGTH
INSERT REMAINING LENGTH
SET CONTINUATION FLAG
DMH60
UPDATE TRACK INDEX
DADIO
WRITE THIS TRACK OF DATA
UPDATE CURRENT TRACK ADDRESS
TRACK ? LAST TIME
YES
NO
INIT OUTPUT FOR NEXT TRACK
INSERT CONT CODE
INIT DMR32
H12
DMH62
UPDATE LAST TRACK INDEX
UPDATE TRACK INDEX POINTER
INITIALIZE TRACK INDEX
DMH60
UPDATE CYL INDEX INDEX
UPDATE CYL INDEX COUNTER
NO
IS CYL INDEX FULL
YES
DMH65
WRITE THIS CYL INDEX ON DISK
INITIALIZE CYL INDEX COUNTER
INCREMENT CYL INDEX POINTER
IS DISK FULL
YES
SVC50
NO
UPDATE CURRENT TRACK POINTER BY 2
DMH60
UPDATE MASTER INDEX
INITIALIZE CYLINDER INDEX
INIT OUTPUT FOR TRACK INDEX
INSERT CONT CODE
INIT DMR32
H12

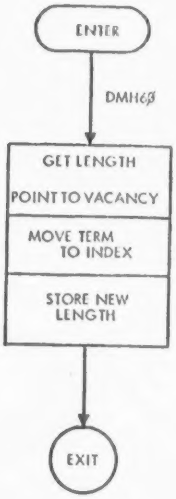
DKDICT

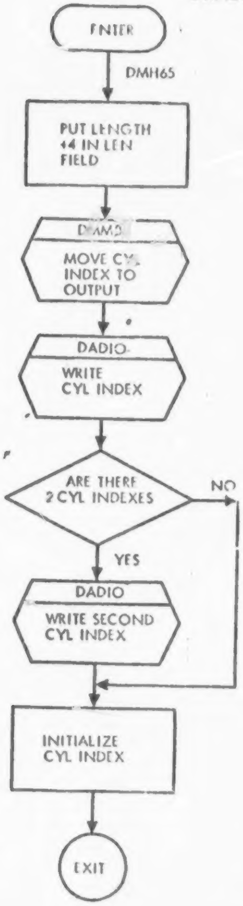

ENTER
INITIALIZE DISK FILE
DMH00
SAVE RETURN
IS DISK SWITCH ON
NO
EXIT
YES
DADIO
FORMAT FIRST TEN TRACKS
INITIALIZE INDEX TABLES COUNTER POINTER
INITIALIZE TRACK 10 (MI, TI, CMV)
DADIO
WRITE TRACK 10
INITIALIZE TRACK 11 RESTORE RETURN
EXIT


DKDICT
ENTER
CLOSE DISK FILE
DMH05
SAVE RETURN
IS DISK SWITCH ON
NO
EXIT
YES
PUT DATE IN COMM VEC GET DUMMY LAST TERM
DMH10
PUT TERM IN DICT
DMH60
UPDATE TRACK, CYL, MASTER INDEX TABLES
DADIO
WRITE LAST TRACK
DMH62
WRITE LAST TRACK INDEX
DMH65
WRITE LAST CYL INDEX
DADIO
READ TRACK 10
INSERT MASTER INDEX AND COMM VECTOR
DADIO
WRITE TRACK 10
RESTORE RETURN
EXIT

DKDICT


ENTER
DMH62
DADIO
READ LAST TRACK INDEX
DMM03
MAKE TRACK INDEX TO OUTPUT
DADIO
WRITE TRACK INDEX
EXIT


ENTER
DMH60
GET LENGTH
POINT TO VACANCY
MOVE TERM TO INDEX
STORE NEW LENGTH
EXIT


ENTER
DMH65
PUT LENGTH +4 IN LEN FIELD
DMM0
MOVE CYL INDEX TO OUTPUT
DADIO
WRITE CYL INDEX
ARE THERE 2 CYL INDEXES
NO
YES
DADIO
WRITE SECOND CYL INDEX
INITIALIZE CYL INDEX
EXIT

DICTIONARY MESSAGE PRINT PROGRAM (DICTMSG) PROCESSING LOGIC

The files are opened and the contents of the UPSI card are scanned to determine which types of messages are to be listed. The proper switches are set and a record is read. If the record is a message the type is checked. If it is to be printed an internal indicator is set and the message is printed. If it is not to be printed the record is ignored. Then the next record is read. As soon as a record is encountered which is not a message (i.e. a dictionary record) the internal indicator is tested. If the message immediately preceding this record has been printed, control is transferred to SBRFORM which reformats and prints the dictionary record. If the messages relating to this record have not been printed the dictionary record is ignored. Then the next record is read and processed. As soon as an end-of-file condition is encountered on the input tape processing terminates.

Input

The tape of messages created by DICT2 is read from SYS008. It contains two types of records: the Message record labeled D3 below and the Dictionary record labeled D2. The format of the tape is variable length records blocked. For a more detailed description of the Dictionary record please refer to Appendix A.

A UPSI control card is used to specify the types of messages and records printed. Its format is shown below:

`//UPSI XXXXXXXX`

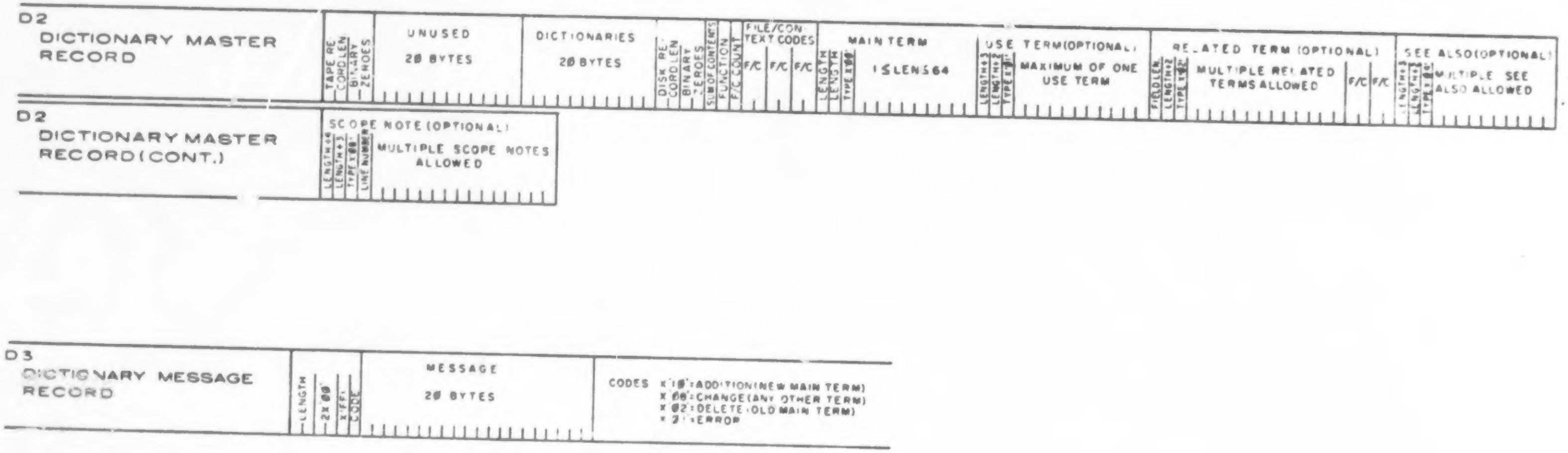
A 1 in position four indicates an addition record will be printed.
A 1 in position five indicates a change record will be printed.
A 1 in position seven indicates a delete record will be printed.
A 1 in position eight indicates an error record will be printed.

A zero (0) in any of the above positions indicates that action is not requested.

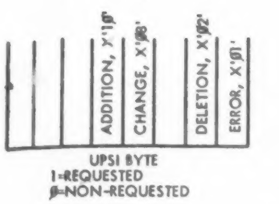
Output

The listing of messages and records is produced by the subroutine SBRFORM. A detailed description of the output format is given in the SBRFORM section.

DICTIONARY MESSAGE PRINT PROGRAM (DICTMSG) RECORD FORMATS


D2
DICTIONARY MASTER RECORD
TAPE RECORDLEN
BINARY ZEROES
UNUSED
20 BYTES
DICTIONARIES
20 BYTES
DISK RECORDLEN
BINARY ZEROES
SUM OF CONTENTS
FUNCTION
F/C COUNT
FILE/CONTEXT CODES
F/C
F/C
F/C
MAIN TERM
LENGTH
LENGTH
TYPE X'00'
1≤LEN≤64
USE TERM(OPTIONAL)
LENGTH+3
LENGTH+2
TYPE X'01'
MAXIMUM OF ONE USE TERM
RELATED TERM (OPTIONAL)
FIELDLEN
LENGTH+2
TYPE X'02'
MULTIPLE RELATED TERMS ALLOWED
F/C
F/C
SEE ALSO(OPTIONAL)
LENGTH+3
LENGTH+2
TYPE X'03'
MULTIPLE SEE ALSO ALLOWED
D2
DICTIONARY MASTER RECORD(CONT.)
SCOPE NOTE (OPTIONAL)
LENGTH+4
LENGTH+3
TYPE X'08'
LINE NUMBER
MULTIPLE SCOPE NOTES ALLOWED
D3
DICTIONARY MESSAGE RECORD
LENGTH
2X'00'
X'FF'
CODE
MESSAGE
20 BYTES
CODES X'10':ADDITION(NEW MAIN TERM)
X'00':CHANGE(ANY OTHER TERM)
X'02':DELETE(OLD MAIN TERM)
X'2':ERROR

DICTMSG
PRINTS DICT2 MESSAGES



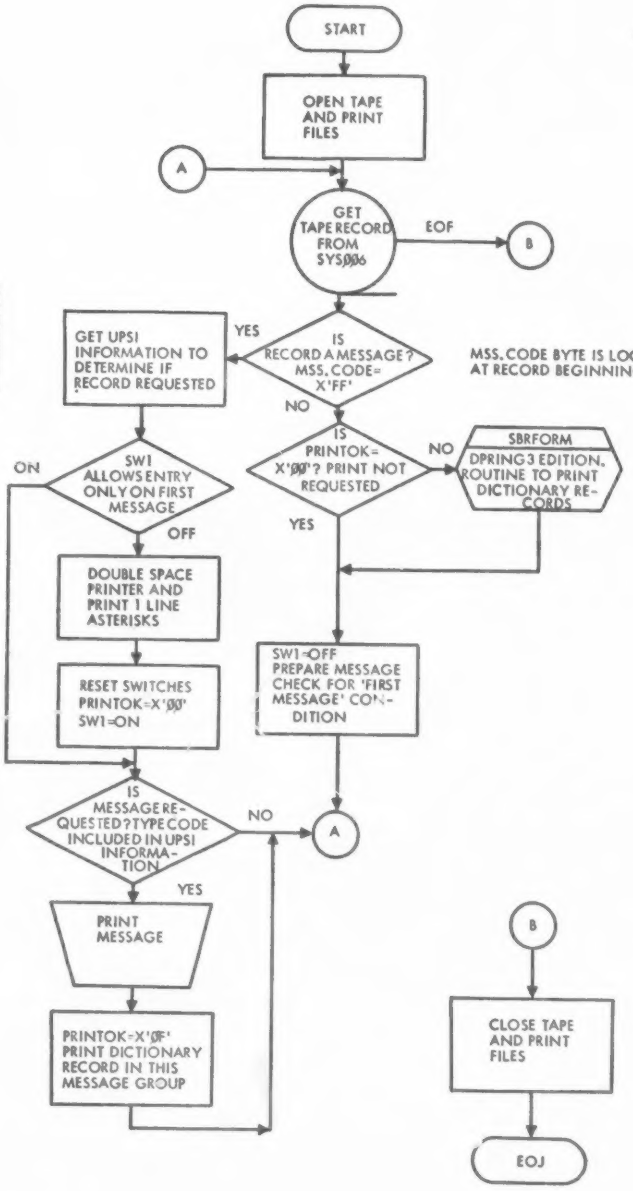
ADDITION, X'10'
CHANGE, X'08'
DELETION, X'02'
ERROR, X'01'

UPSI BYTE
1=REQUESTED
0=NON-REQUESTED

PRINTOK=ALLOWS DICT RECORDS TO BE PRINTED.

TYPE CODE LOCATED AT RECORD BEGINNING +5

TYPE CODES:
X'10'=ADDITION(NEW MAIN TERM)
X'08'=CHANGE (ANY OTHER TERM)
X'02'=DELETION(OLD MAIN TERM)
X'01'=ERROR



START
OPEN TAPE AND PRINT FILES
A
GET TAPE RECORD FROM SYS006
EOF
B
IS RECORD A MESSAGE? MSS. CODE= X'FF'
YES
NO
GET UPSI INFORMATION TO DETERMINE IF RECORD REQUESTED
IS PRINTOK= X'00'? PRINT NOT REQUESTED
NO
YES
SBRFORM
DPRING3 EDITION. ROUTINE TO PRINT DICTIONARY RECORDS
SW1 ALLOWS ENTRY ONLY ON FIRST MESSAGE
ON
OFF
DOUBLE SPACE PRINTER AND PRINT 1 LINE ASTERISKS
RESET SWITCHES PRINTOK=X'00' SW1=ON
SW1=OFF PREPARE MESSAGE CHECK FOR 'FIRST MESSAGE' CONDITION
IS MESSAGE REQUESTED? TYPE CODE INCLUDED IN UPSI INFORMATION
NO
A
YES
PRINT MESSAGE
PRINTOK=X'0F' PRINT DICTIONARY RECORD IN THIS MESSAGE GROUP

MSS. CODE BYTE IS LOCATED AT RECORD BEGINNING +4

B
CLOSE TAPE AND PRINT FILES
EOJ

DICTIONARY PRINT PROGRAM (DPRINT3) PROCESSING LOGIC

DP000 Main Control

All switches and counters are first initialized. Input/output areas and work areas are cleared.

The main program transfers control to subroutine DPB00 which reads and edit checks the request cards. If there are no valid requests, all files are closed and the job is terminated.

If there are valid requests a test is made to see if a dump has been requested. If there is not a dump request, the valid requests are sorted in ascending order.

The main program then transfers control to subroutine DPE00 which controls the printing of the report.

All files are closed and the job is terminated.

DPB00 Read Data Cards

This subroutine reads cards until either an end of file is read or until the input buffer size is exceeded. A maximum of one hundred data cards are allowed. When either of these conditions occurs, a message is printed and control is returned to the main routine. Each card is given a sequence number, and then control is transferred to subroutine DPC00 to edit check and reformat the card.

If the card is in error, the sequence number, card image, and an error message are printed.

If the card passes all edit checks, the sequence number and card image are printed. Then the reformatted card image is moved into the next 68 bytes in the request buffer in core. The pointer to the request buffer entry and the number of valid terms in the buffer are incremented.

If the first data card was not a "dump" request card, the subroutine reads the next card.

If the first data card was a "dump" request card, a message is printed stating that no more cards will be processed.

Any cards in the card reader are passed through until a system "end of file" card is read. Then control is returned to the main routine.

DPC00 Edit and Reformat Cards

The reformatted card buffer and the pointer for the card image are initialized. The card is scanned from the left a column at a time. Blanks between terms are ignored.

If the first character encountered is a "T", the card is assumed to be a term request. Columns are skipped until a slash is encountered. The next character encountered is considered the beginning of the main term name. Characters are moved to the reformatted card area until a semi-colon is found; multiple blanks are condensed down to one blank. The length of the term is incremented by one and inserted in the reformatted card area.

Columns are again skipped until a slash is encountered. The next character encountered is assumed to be the first character of a parameter request.

Control is transferred to subroutine DPO00, which compares the parameter against the following fields and indicates whether the parameter is valid or not, and, if valid, sets the appropriate indicator.

Field	Description
RECORD	all of the options
AN	authority of nomenclature
DICT	dictionaries
FN	functions
FC	file context codes
SCOPE	scope notes
SA	see also references
RT	related terms
USE	use term

Then control is returned to this routine (DPC00). If a comma follows the parameter, the program expects another parameter which is compared in the same manner.

If the end of card is detected following a parameter, the card is a valid request. The indicator for printing the use term is turned on, and control is returned to the calling routine.

If the first character encountered in the card is not a "T" and if this is the first data card, and if the first character is a "D", then the card is assumed to be a dump request. Columns are skipped until a slash is encountered. Then the card is scanned for parameters by exactly the same method as the term card.

If the first character is other than described above, or if an error is detected in the card, or if the end of card is reached before the edit checks have all been performed; an error indication is given and control is returned to the calling routine.

The format of each reformatted card is as follows:

Bits 0-7

AN DICT FN FC SCOPE SA RT USE

term length	main term		card seq number	unused
1 byte Byte No. (1)	64 bytes (2-65)	1 byte (66)	1 byte (67)	1 byte (68)

Where a "1" in any bit position of byte 66 indicates that the appropriate portion of the main term is to be printed.

If there are no valid requests at this point, control is returned to the calling program.

If there are valid request entries, pointers and switches are initialized and the number of requests is interrogated. If there are no requests remaining, a message is printed and control is returned to the calling program.

The first request is tested to see if it is a "dump" request. If not, it is a "term" request and is processed as one. If it is a "dump" request, the program reads the next dictionary record. An end of file condition causes the program to print a message and return control to the calling program. If the record is not an end of file record, control is transferred to SBRFORM which does the actual printing of the record. After control is returned, this routine continues reading dictionary records and printing until an end of file condition is encountered.

If the request is a "term" request, the request main term is compared to the current dictionary main term. If the dictionary main term compares high, the term requested is not on the dictionary tape. A message is printed along with the main term. The program gets another request.

If the dictionary main term compares low, the program reads another dictionary record. If it is not an end of file record, the program goes back to compare again. If it is an end of file record, a message is printed stating that the requested main term is not on the dictionary tape. The program gets another request.

If the dictionary main term equals the request main term, the program transfers control to routine SBRFORM which prints the record. When control is returned, this program gets another request.

SBRFORM Subroutine

SBRFORM formats and prints the data in dictionary records read into the calling program. For a detailed description of the logic and format for SBRFORM refer to that section within this manual.

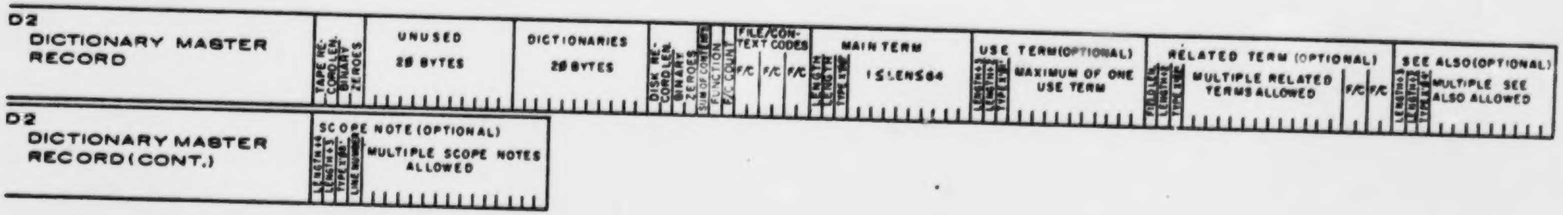
Input

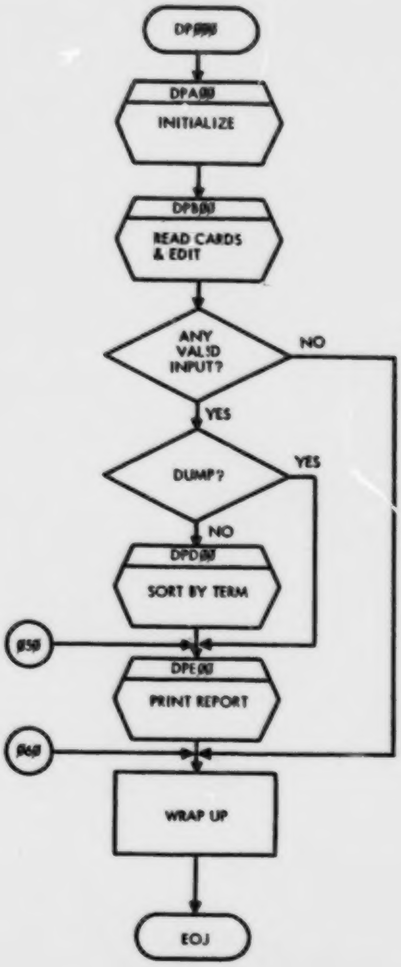
The dictionary tape is read on SYS007. Its format is labelled D2 below. For a more detailed description of the dictionary master record please refer to Appendix A. In addition to the input tape, control cards which influence the printing are read on SYSRDR. These are either DUMP (dump the entire tape) or TERM (selective processing) cards. The secondary parameter field specifies the amount of the record to be printed.

Output

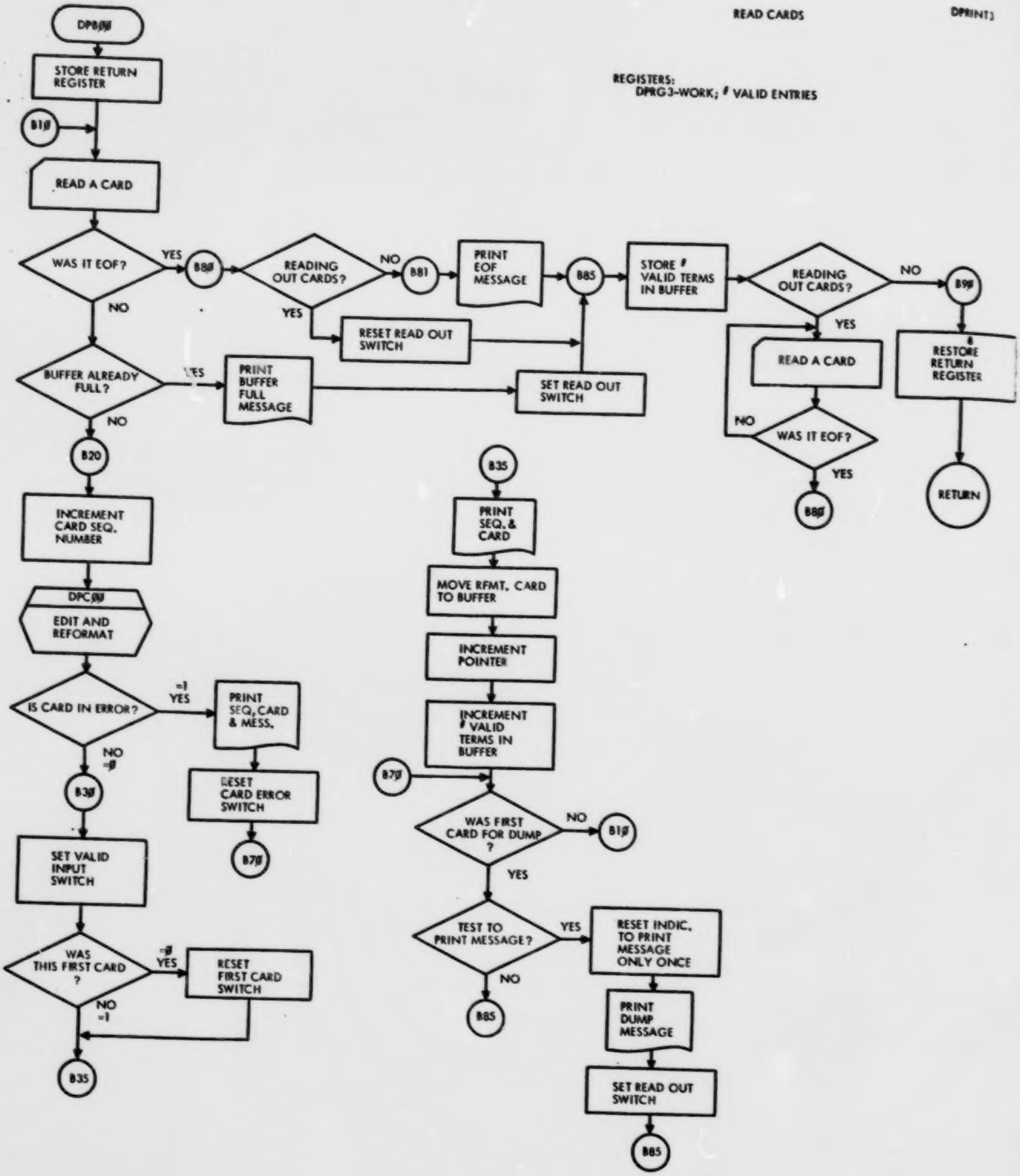
The subroutine SBRFORM lists the dictionary in the format described in detail in the SBRFORM section of this manual.

DICTIONARY PRINT PROGRAM (DPRINT3) RECORD FORMATS


D2
DICTIONARY MASTER RECORD
TAPE RE-CORDLEN.
BINARY ZEROES
UNUSED
20 BYTES
DICTIONARIES
20 BYTES
DISK RE-CORDLEN.
BINARY ZEROES
SUM OF CONT.
FUNCTION
F/C COUNT
FILE/CONTEXT CODES
F/C
F/C
F/C
MAIN TERM
LENGTH
LENGTH
1≤LEN≤64
USE TERM(OPTIONAL)
MAXIMUM OF ONE USE TERM
RELATED TERM (OPTIONAL)
MULTIPLE RELATED TERMS ALLOWED
F/C
F/C
SEE ALSO(OPTIONAL)
MULTIPLE SEE ALSO ALLOWED
D2
DICTIONARY MASTER RECORD(CONT.)
SCOPE NOTE (OPTIONAL)
LINE NUMBER
MULTIPLE SCOPE NOTES ALLOWED


DP000
DPA00
INITIALIZE
DPB00
READ CARDS & EDIT
ANY VALID INPUT?
NO
YES
DUMP?
YES
NO
DPD00
SORT BY TERM
050
DPE00
PRINT REPORT
060
WRAP UP
EOJ

REGISTERS:
DPRG3-WORK; # VALID ENTRIES

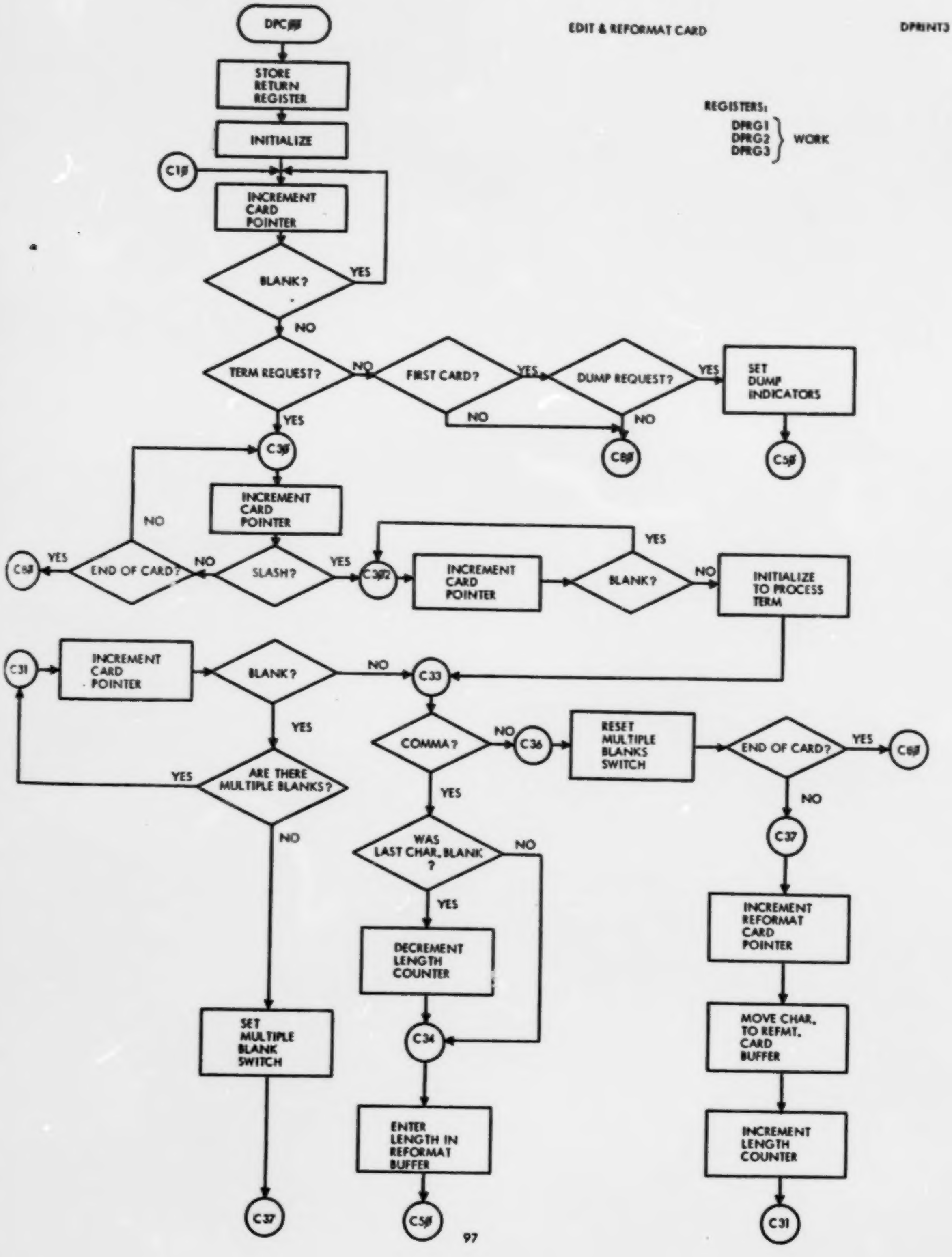

DPB00
STORE RETURN REGISTER
B10
READ A CARD
WAS IT EOF?
YES
NO
B80
READING OUT CARDS?
NO
YES
B81
PRINT EOF MESSAGE
RESET READ OUT SWITCH
B85
STORE # VALID TERMS IN BUFFER
READING OUT CARDS?
NO
YES
B90
READ A CARD
WAS IT EOF?
NO
YES
B80
RESTORE RETURN REGISTER
RETURN
BUFFER ALREADY FULL?
YES
NO
PRINT BUFFER FULL MESSAGE
SET READ OUT SWITCH
B20
INCREMENT CARD SEQ. NUMBER
DPC00
EDIT AND REFORMAT
IS CARD IN ERROR?
=1
YES
NO
=0
PRINT SEQ, CARD & MESS.
RESET CARD ERROR SWITCH
B70
B30
SET VALID INPUT SWITCH
WAS THIS FIRST CARD?
=0
YES
NO
=1
RESET FIRST CARD SWITCH
B35
B35
PRINT SEQ, & CARD
MOVE RFMT. CARD TO BUFFER
INCREMENT POINTER
INCREMENT # VALID TERMS IN BUFFER
B70
WAS FIRST CARD FOR DUMP?
NO
B10
YES
TEST TO PRINT MESSAGE?
YES
NO
B85
RESET INDIC. TO PRINT MESSAGE ONLY ONCE
PRINT DUMP MESSAGE
SET READ OUT SWITCH
B85

EDIT & REFORMAT CARD

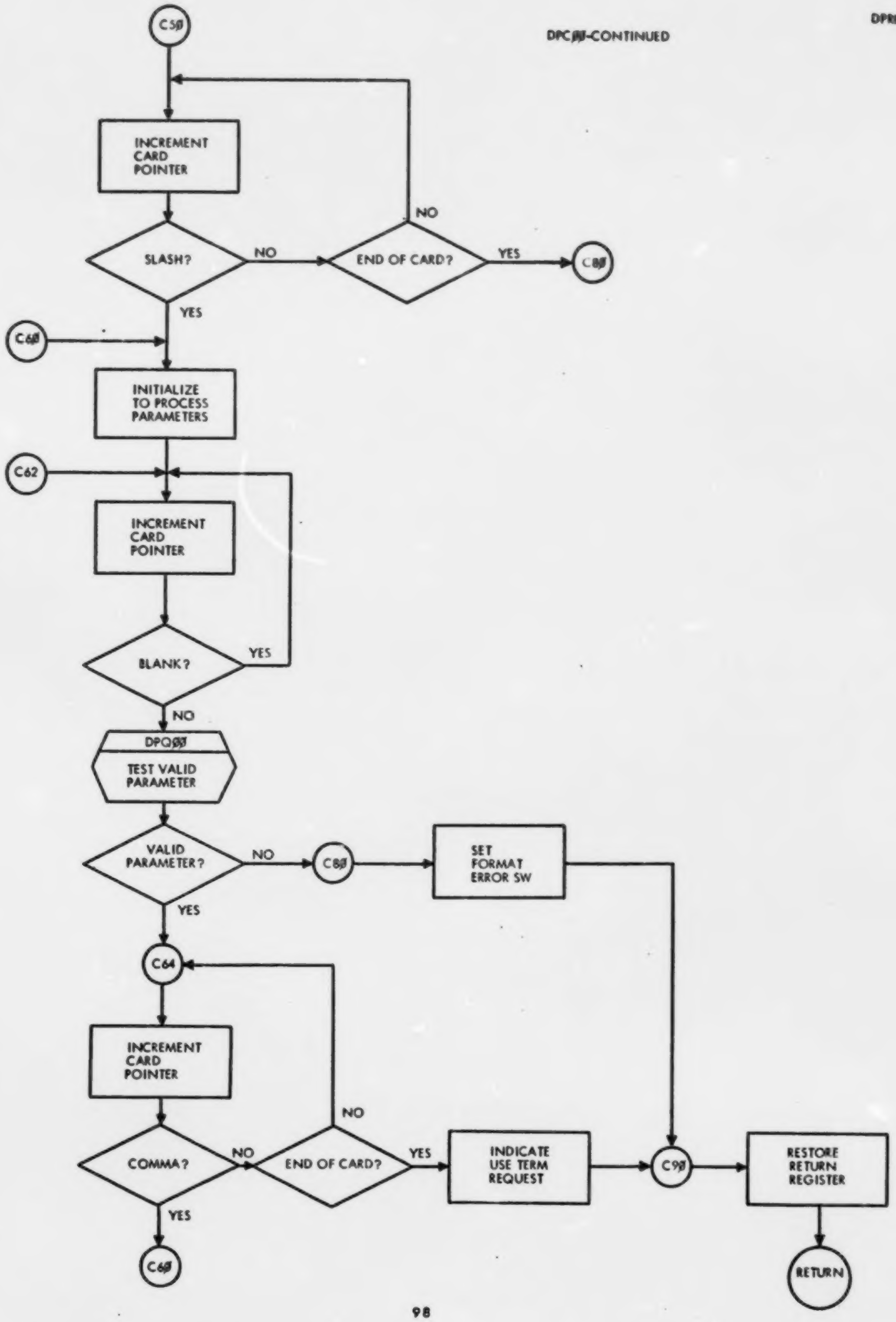
DPRINT3

REGISTERS:

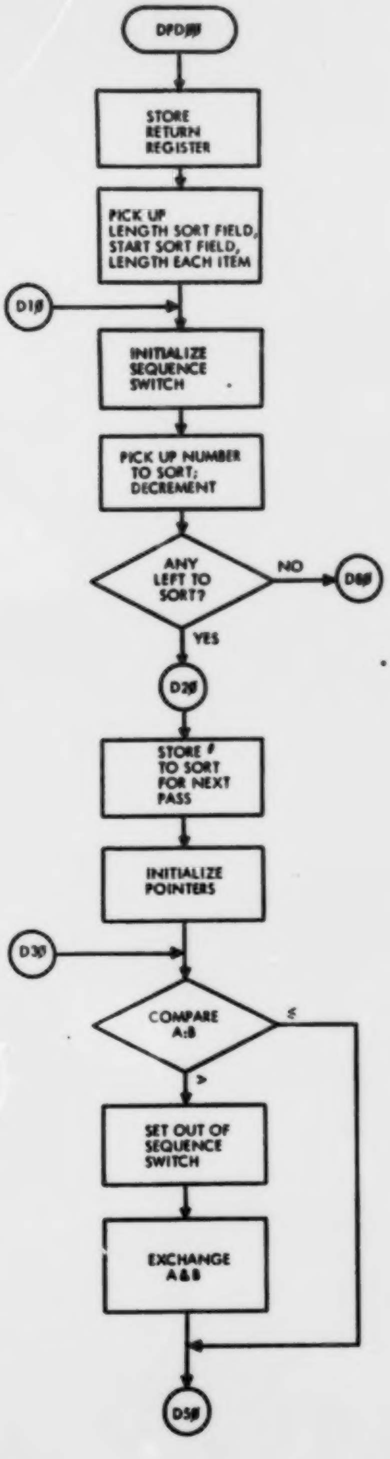
DPRG1 } WORK
DPRG2
DPRG3


DPC00
STORE RETURN REGISTER
INITIALIZE
C10
INCREMENT CARD POINTER
BLANK?
YES
NO
TERM REQUEST?
NO
FIRST CARD?
YES
DUMP REQUEST?
YES
SET DUMP INDICATORS
NO
NO
YES
C30
C80
C50
INCREMENT CARD POINTER
NO
YES
C60
END OF CARD?
NO
SLASH?
YES
C302
INCREMENT CARD POINTER
BLANK?
YES
NO
INITIALIZE TO PROCESS TERM
C31
INCREMENT CARD POINTER
BLANK?
NO
C33
YES
ARE THERE MULTIPLE BLANKS?
YES
NO
COMMA?
NO
C36
RESET MULTIPLE BLANKS SWITCH
END OF CARD?
YES
C60
NO
C37
YES
WAS LAST CHAR. BLANK ?
NO
YES
DECREMENT LENGTH COUNTER
C34
ENTER LENGTH IN REFORMAT BUFFER
C50
SET MULTIPLE BLANK SWITCH
C37
INCREMENT REFORMAT CARD POINTER
MOVE CHAR. TO REFMT. CARD BUFFER
INCREMENT LENGTH COUNTER
C31

DPC00-CONTINUED

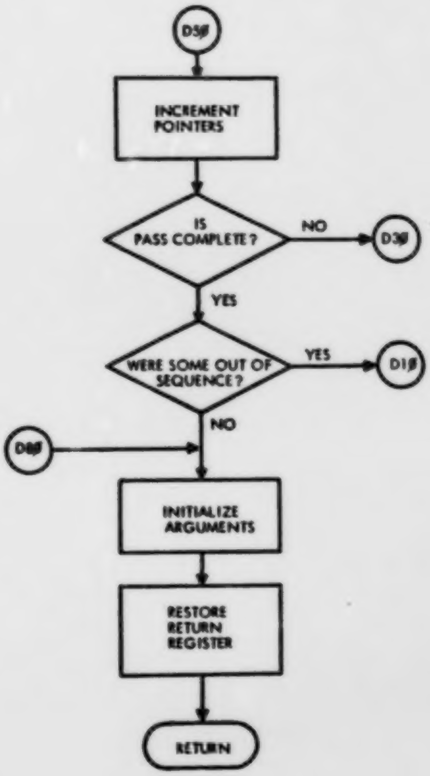

C50
INCREMENT CARD POINTER
SLASH?
NO
END OF CARD?
NO
YES
C80
YES
C60
INITIALIZE TO PROCESS PARAMETERS
C62
INCREMENT CARD POINTER
BLANK?
YES
NO
DPQ00
TEST VALID PARAMETER
VALID PARAMETER?
NO
C80
SET FORMAT ERROR SW
YES
C64
INCREMENT CARD POINTER
COMMA?
NO
END OF CARD?
NO
YES
INDICATE USE TERM REQUEST
C90
RESTORE RETURN REGISTER
YES
C60
RETURN

INTERNAL SORT


DPD00
STORE RETURN REGISTER
PICK UP LENGTH SORT FIELD, START SORT FIELD, LENGTH EACH ITEM
D10
INITIALIZE SEQUENCE SWITCH
PICK UP NUMBER TO SORT; DECREMENT
ANY LEFT TO SORT?
NO
D80
YES
D20
STORE # TO SORT FOR NEXT PASS
INITIALIZE POINTERS
D30
COMPARE A:B
≤
>
SET OUT OF SEQUENCE SWITCH
EXCHANGE A&B
D50

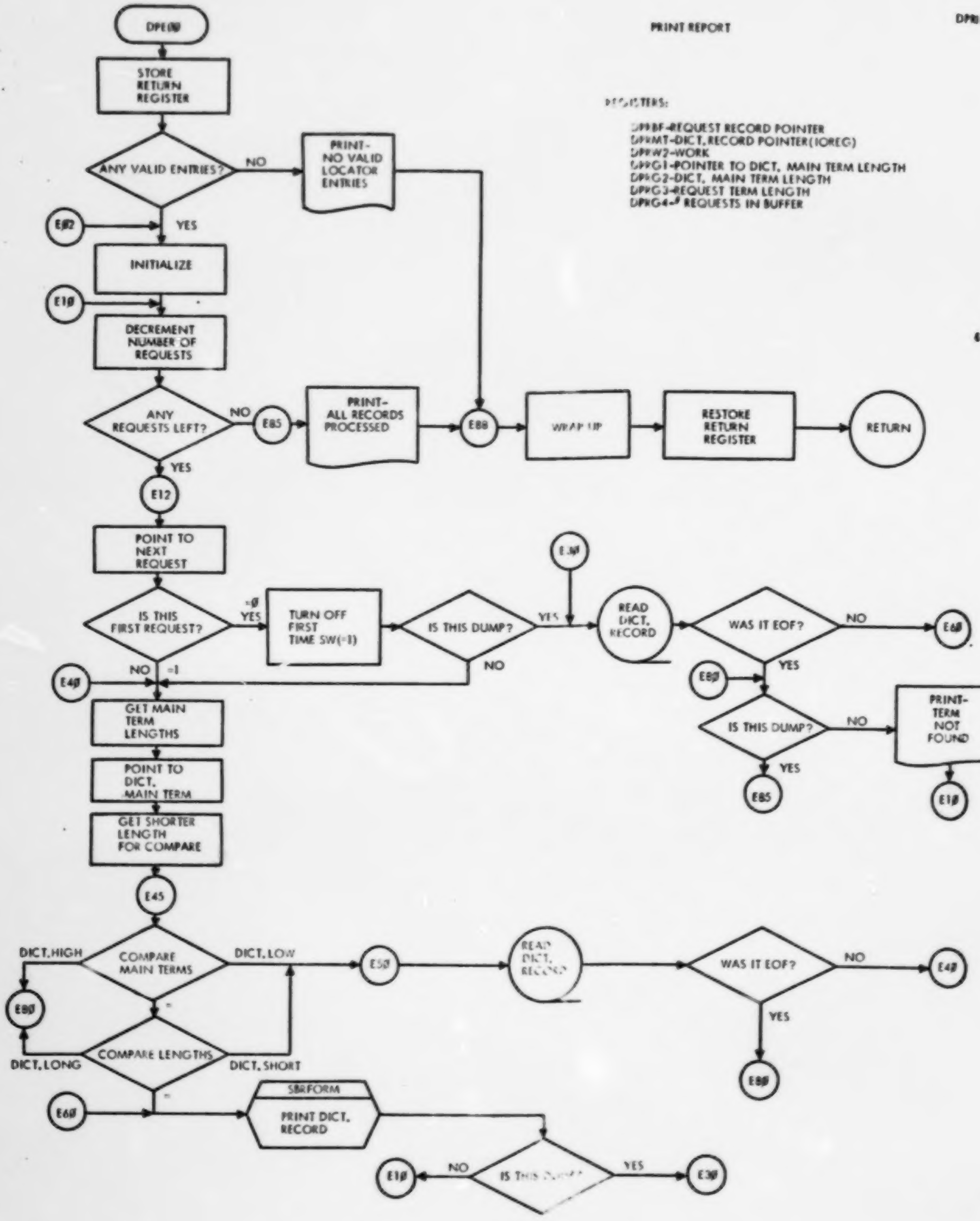
BEFORE BRANCHING TO SORT ROUTINE:
DPDA1=START OF DATA TO SORT (F)
DPDA2=NUMBER OF ITEMS TO SORT (H)
DPDA3=START OF SORT FIELD (X)
DPDA4=LENGTH OF EACH ITEM (X)
DPDA5=LENGTH OF SORT FIELD (X)

REGISTERS:
DPRW1 }
DPRW2 }
DPRG1 } WORK REGISTERS
DPRG2 }
DPRG3 }


D50
INCREMENT POINTERS
IS PASS COMPLETE?
NO
D30
YES
WERE SOME OUT OF SEQUENCE?
YES
D10
NO
D80
INITIALIZE ARGUMENTS
RESTORE RETURN REGISTER
RETURN

PRINT REPORT

DPRINT3

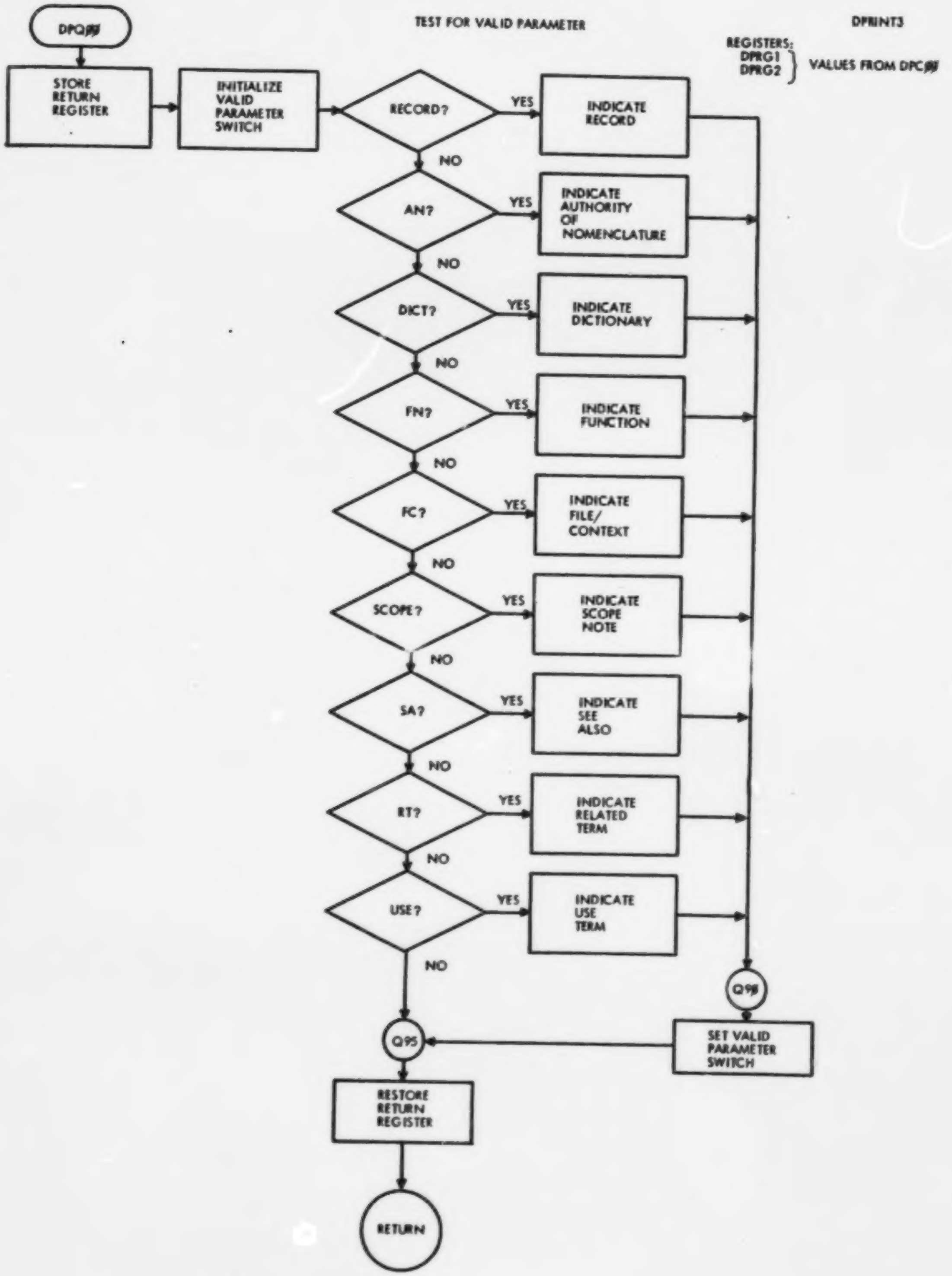

DPE00
STORE RETURN REGISTER
ANY VALID ENTRIES?
NO
PRINT- NO VALID LOCATOR ENTRIES
E02
YES
INITIALIZE
E10
DECREMENT NUMBER OF REQUESTS
ANY REQUESTS LEFT?
NO
E85
PRINT- ALL RECORDS PROCESSED
E88
WRAP UP
RESTORE RETURN REGISTER
RETURN
YES
E12
POINT TO NEXT REQUEST
E30
IS THIS FIRST REQUEST?
=0 YES
TURN OFF FIRST TIME SW(=1)
IS THIS DUMP?
YES
READ DICT. RECORD
WAS IT EOF?
NO
E60
YES
NO
=1
NO
E40
E80
IS THIS DUMP?
NO
PRINT- TERM NOT FOUND
YES
E85
E10
GET MAIN TERM LENGTHS
POINT TO DICT. MAIN TERM
GET SHORTER LENGTH FOR COMPARE
E45
DICT. HIGH
COMPARE MAIN TERMS
DICT. LOW
E50
READ DICT. RECORD
WAS IT EOF?
NO
E40
YES
E80
E80
=
DICT. LONG
COMPARE LENGTHS
DICT. SHORT
=
E60
SBRFORM
PRINT DICT. RECORD
E10
NO
IS THIS DUMP?
YES
E30

REGISTERS:

DPRBF-REQUEST RECORD POINTER
DPRMT-DICT. RECORD POINTER(IOREG)
DPRW2-WORK
DPRG1-POINTER TO DICT. MAIN TERM LENGTH
DPRG2-DICT. MAIN TERM LENGTH
DPRG3-REQUEST TERM LENGTH
DPRG4-# REQUESTS IN BUFFER

TEST FOR VALID PARAMETER

DPRINT3


DPQ00
REGISTERS:
DPRG1
DPRG2
VALUES FROM DPC00
STORE RETURN REGISTER
INITIALIZE VALID PARAMETER SWITCH
RECORD?
YES
INDICATE RECORD
NO
AN?
YES
INDICATE AUTHORITY OF NOMENCLATURE
NO
DICT?
YES
INDICATE DICTIONARY
NO
FN?
YES
INDICATE FUNCTION
NO
FC?
YES
INDICATE FILE/ CONTEXT
NO
SCOPE?
YES
INDICATE SCOPE NOTE
NO
SA?
YES
INDICATE SEE ALSO
NO
RT?
YES
INDICATE RELATED TERM
NO
USE?
YES
INDICATE USE TERM
NO
Q90
SET VALID PARAMETER SWITCH
Q95
RESTORE RETURN REGISTER
RETURN

DICTIONARY RECORD PRINT SUBROUTINE (SBRFORM)

Purpose

SBRFORM is a subroutine which is called by DPRINT3 and DICTMSG. Upon entry is tests for the presence of a main term. If none is present a message is printed in place of the main term and the rest of the record is processed. If a main term is encountered it is double printed (for emphasis) and the subordinate data is processed. Since the printing of subordinate data may or may not be requested (by DPRINT3) each type of data encountered is tested against an internal switch. If the type of data being processed is requested it is titled and printed out.

SBRFORM processes the following types of subordinate data: dictionary entries, authority of nomenclature (no longer used but the logic is still included), file and context codes, function of main term, use terms, related terms (with their own file/context codes), see also terms, and scope notes. The function code is converted to a two character mnemonic prior to printing. The scope notes are printed along with their corresponding line numbers.

As each data type is encountered, a switch is tested to determine whether or not this type has been encountered before in this record. If it has not, a title is printed and the term is processed. Multiple terms of the same type may occur on one line. Scope notes are always placed on separate lines. As soon as the end of the logical record is reached, control is returned to the calling program.

Input

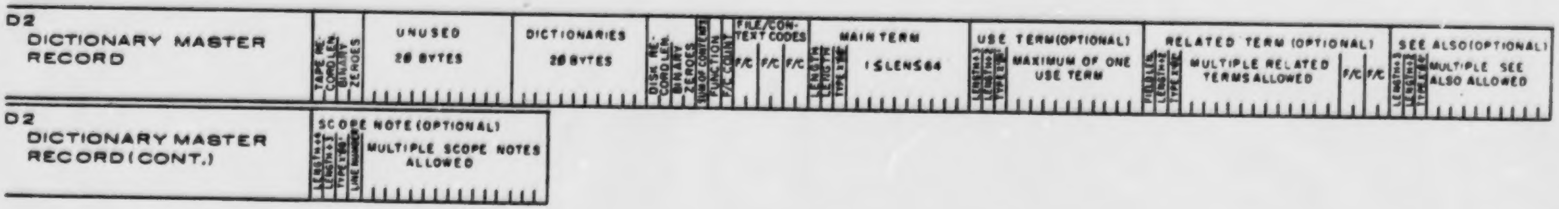
The dictionary master record, labelled D2 below, is reformatted and listed by SBRFORM. For a more detailed description of the record please refer to Appendix A. In addition to the record, parameters are passed from the calling program which specify the types of data to be printed from the record.

Output

Dictionary records will be printed in the following format:

```
(main term)                 (double printed)

AUTHORITY--                 (auth.)
FUNCTION                    (fn)
DICTIONARIES                (dictionary nos.)
FILE/CONTEXT NUMBERS--      (f-c, f-c, f-c, f-c,...)
USE--                       (term)  (double printed)  (term to be used in
                            lieu of main term)
or
RELATED TERMS--             n (where n is the no. of related terms in the
                            record)  (rt1, rt2, rt3, rt4, rt5,... rtn-1, rtn)
SEE ALSO--                  (double printed)
                            (sa1, sa2, sa3,...)
SCOPE NOTES--
                            line no. n (text)
```

DICTIONARY RECORD PRINT SUBROUTINE (SBRFORM) RECORD FORMAT


D2
DICTIONARY MASTER RECORD
TAPE RE-CORD LEN.
BINARY ZEROES
UNUSED
2Ø BYTES
DICTIONARIES
2Ø BYTES
DISK RE-CORD LEN.
BINARY ZEROES
SUM OF CONTENTS
FUNCTION
F/C COUNT
FILE/CON-TEXT CODES
F/C
F/C
F/C
MAIN TERM
LENGTH
LENGTH
TYPE CODE
1≤LEN≤64
USE TERM(OPTIONAL)
LENGTH+3
LENGTH+2
TYPE CODE
MAXIMUM OF ONE USE TERM
RELATED TERM (OPTIONAL)
FIELD LEN.
LENGTH+2
TYPE CODE
MULTIPLE RELATED TERMS ALLOWED
F/C
F/C
SEE ALSO(OPTIONAL)
LENGTH+3
LENGTH+2
TYPE CODE
MULTIPLE SEE ALSO ALLOWED
D2
DICTIONARY MASTER RECORD(CONT.)
SCOPE NOTE (OPTIONAL)
LENGTH+4
LENGTH+3
TYPE CODE
LINE NUMBER
MULTIPLE SCOPE NOTES ALLOWED

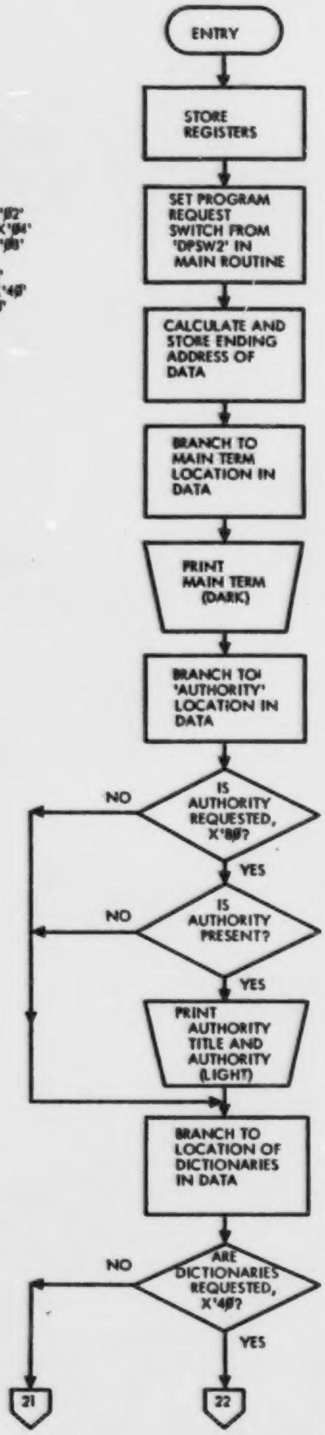
SELECTIVE LIST SUBROUTINE

PRINT DARK=2x (OVERPRINT)
PRINT LIGHT=1x

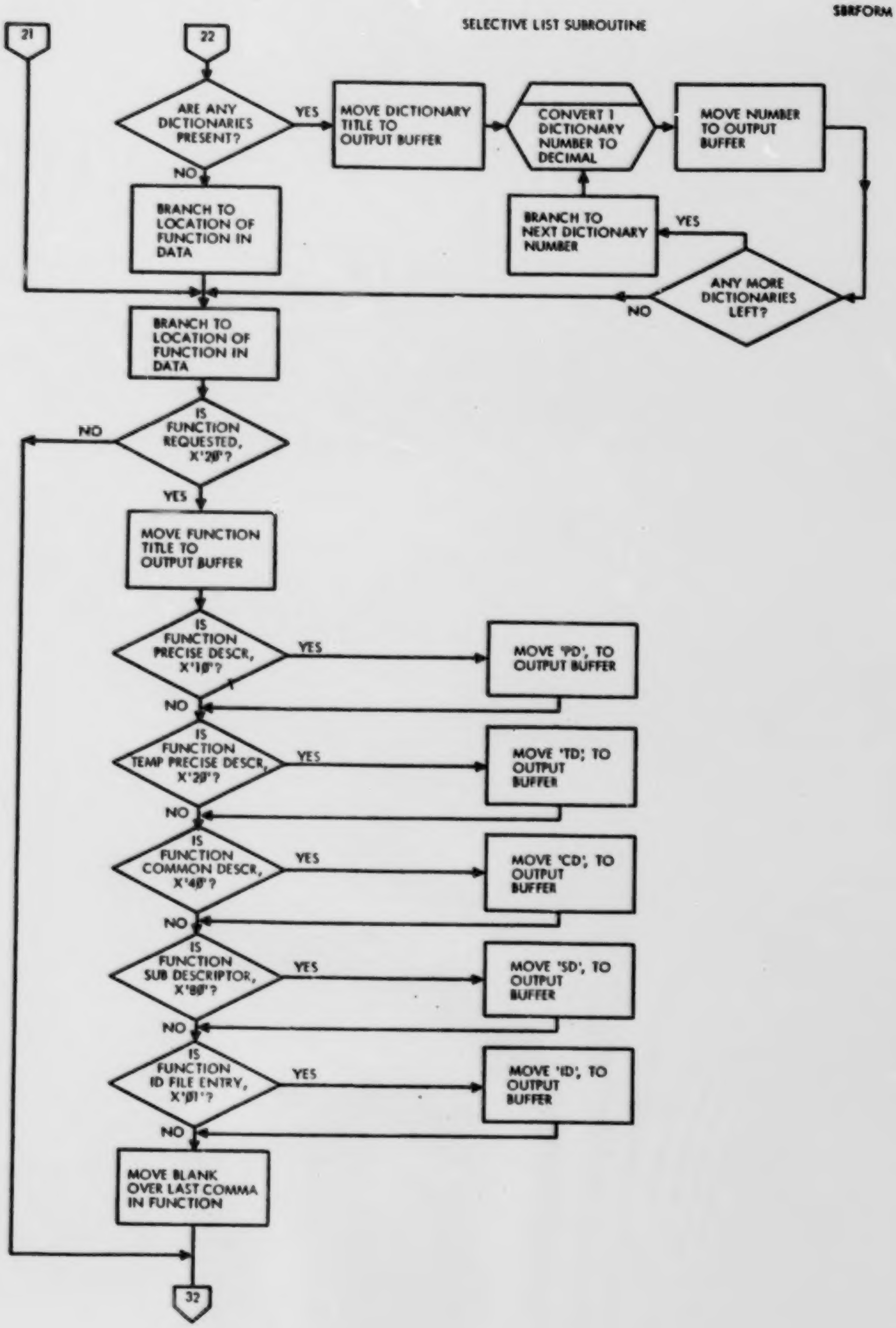
SBRFORM
DPRINT 3 EDITION

DPSW2 SETTINGS

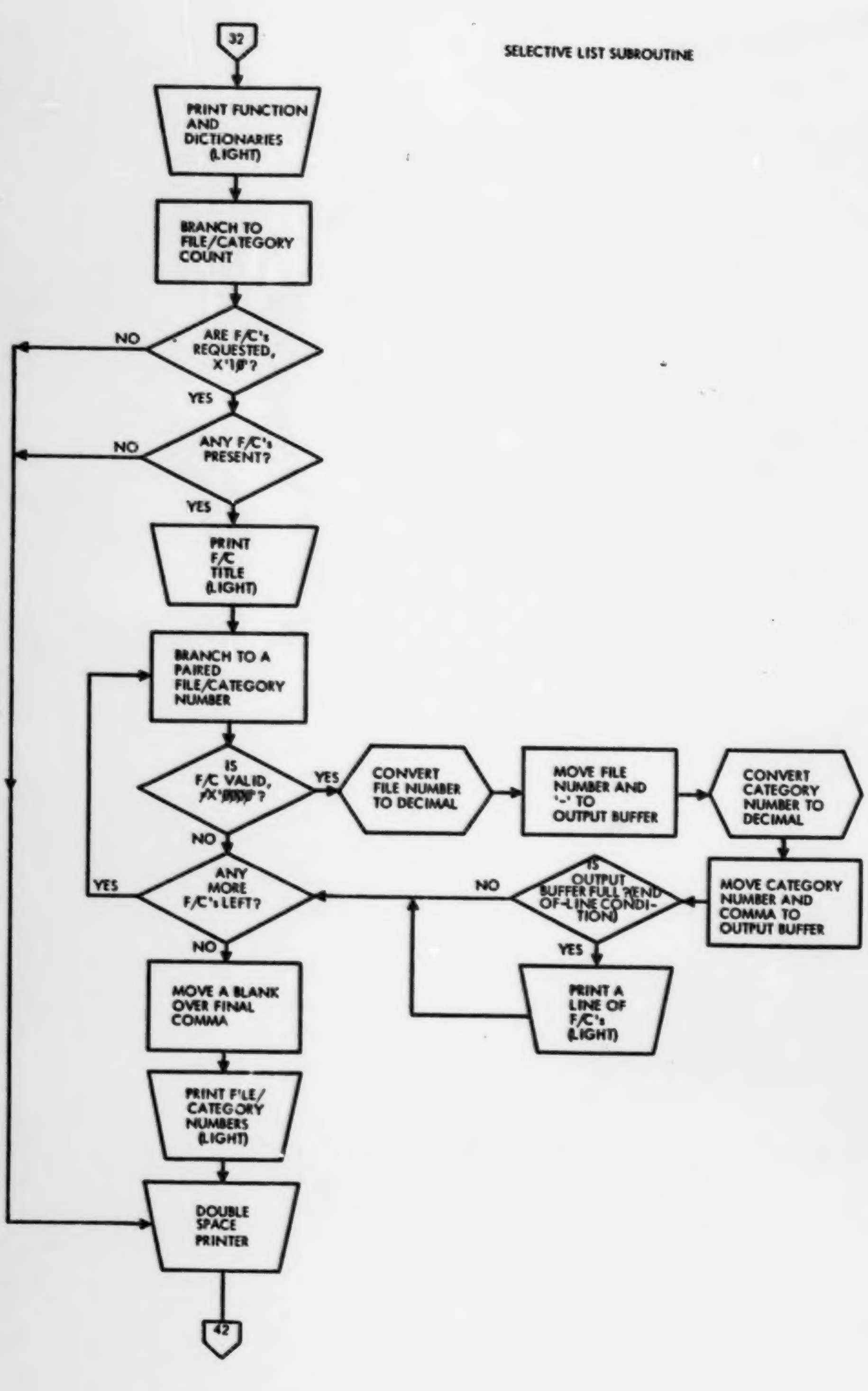
LIST-USE TERM=X'01'
LIST-RELATED TERM=X'02'
LIST-SEE ALSO TERM=X'04'
LIST-SCOPE NOTES=X'08'
LIST-FILE/CATS=X'10'
LIST-FUNCTION=X'20'
LIST-DICTIONARIES=X'40'
LIST-AUTHORITY=X'80'


ENTRY
STORE REGISTERS
SET PROGRAM REQUEST SWITCH FROM 'DPSW2' IN MAIN ROUTINE
CALCULATE AND STORE ENDING ADDRESS OF DATA
BRANCH TO MAIN TERM LOCATION IN DATA
PRINT MAIN TERM (DARK)
BRANCH TO 'AUTHORITY' LOCATION IN DATA
IS AUTHORITY REQUESTED, X'80'?
NO
YES
IS AUTHORITY PRESENT?
NO
YES
PRINT AUTHORITY TITLE AND AUTHORITY (LIGHT)
BRANCH TO LOCATION OF DICTIONARIES IN DATA
ARE DICTIONARIES REQUESTED, X'40'?
NO
YES
21
22

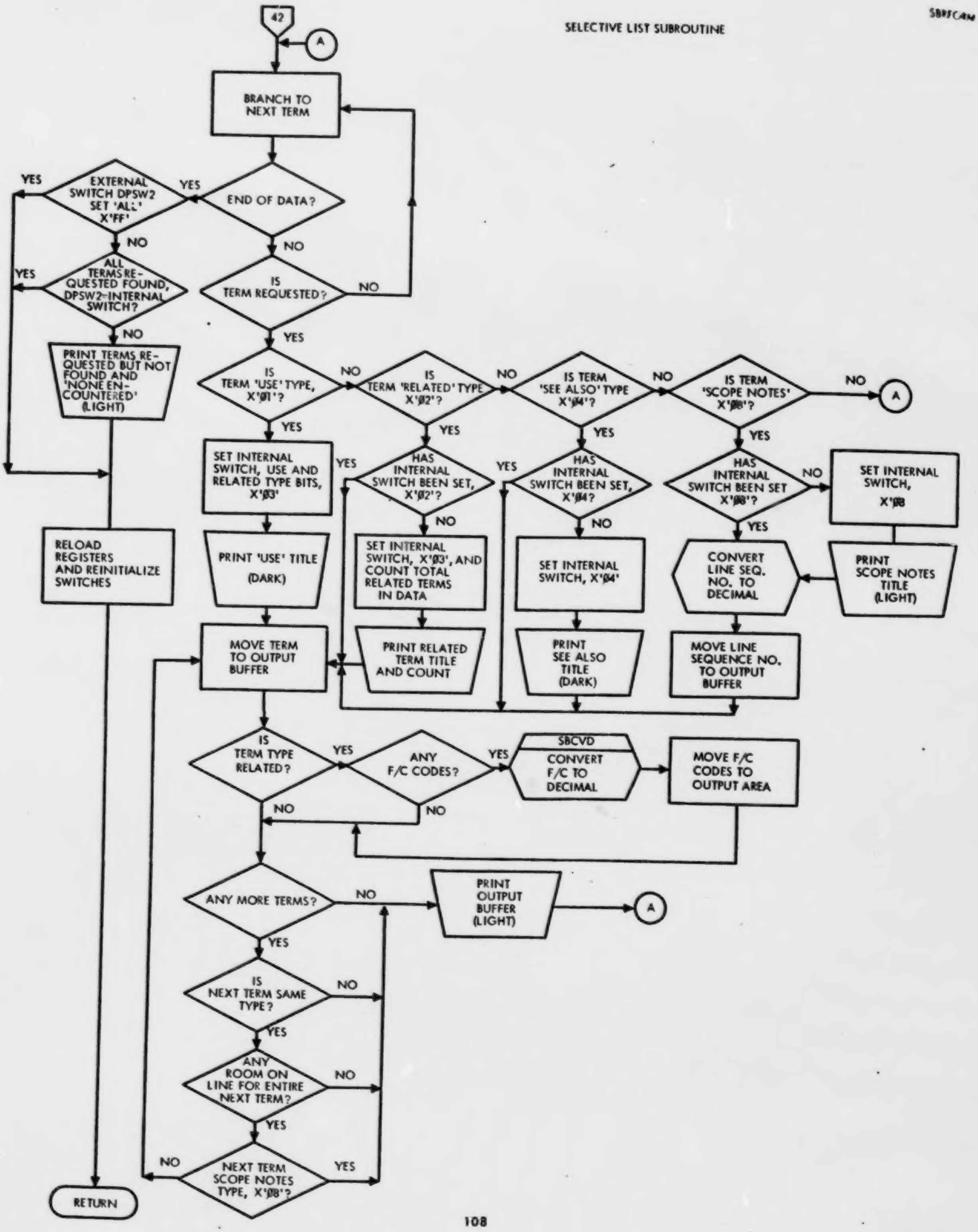
SELECTIVE LIST SUBROUTINE


21
22
ARE ANY DICTIONARIES PRESENT?
YES
MOVE DICTIONARY TITLE TO OUTPUT BUFFER
CONVERT 1 DICTIONARY NUMBER TO DECIMAL
MOVE NUMBER TO OUTPUT BUFFER
NO
BRANCH TO LOCATION OF FUNCTION IN DATA
BRANCH TO NEXT DICTIONARY NUMBER
YES
ANY MORE DICTIONARIES LEFT?
NO
BRANCH TO LOCATION OF FUNCTION IN DATA
NO
IS FUNCTION REQUESTED, X'20'?
YES
MOVE FUNCTION TITLE TO OUTPUT BUFFER
IS FUNCTION PRECISE DESCR, X'10'?
YES
MOVE 'PD', TO OUTPUT BUFFER
NO
IS FUNCTION TEMP PRECISE DESCR, X'20'?
YES
MOVE 'TD', TO OUTPUT BUFFER
NO
IS FUNCTION COMMON DESCR, X'40'?
YES
MOVE 'CD', TO OUTPUT BUFFER
NO
IS FUNCTION SUB DESCRIPTOR, X'80'?
YES
MOVE 'SD', TO OUTPUT BUFFER
NO
IS FUNCTION ID FILE ENTRY, X'01'?
YES
MOVE 'ID', TO OUTPUT BUFFER
NO
MOVE BLANK OVER LAST COMMA IN FUNCTION
32

SELECTIVE LIST SUBROUTINE

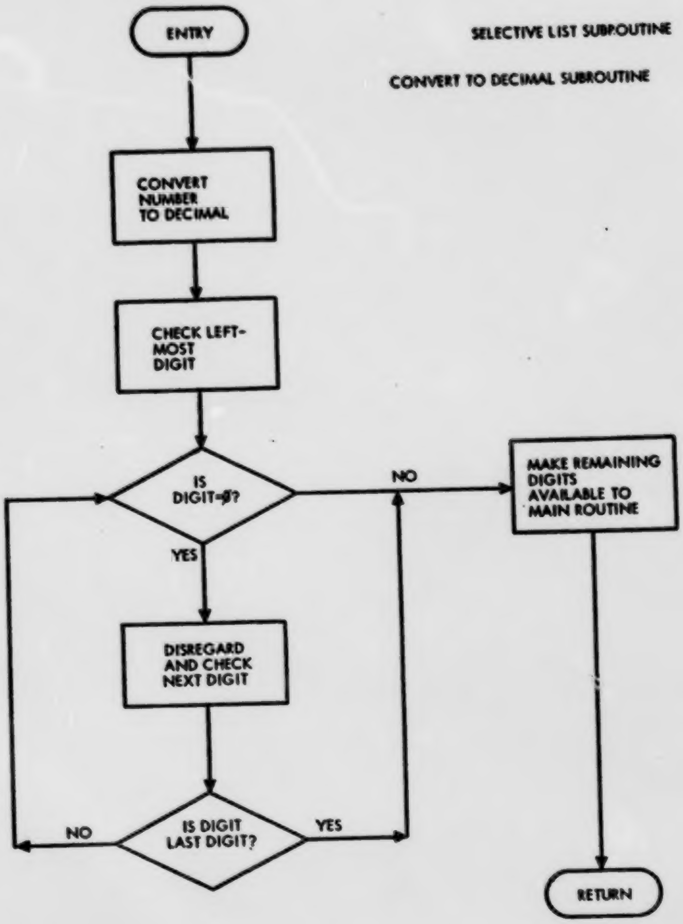

32
PRINT FUNCTION AND DICTIONARIES (LIGHT)
BRANCH TO FILE/CATEGORY COUNT
ARE F/C's REQUESTED, X'10'?
NO
YES
ANY F/C's PRESENT?
NO
YES
PRINT F/C TITLE (LIGHT)
BRANCH TO A PAIRED FILE/CATEGORY NUMBER
IS F/C VALID, /X'[illegible]'?
YES
NO
CONVERT FILE NUMBER TO DECIMAL
MOVE FILE NUMBER AND '-' TO OUTPUT BUFFER
CONVERT CATEGORY NUMBER TO DECIMAL
MOVE CATEGORY NUMBER AND COMMA TO OUTPUT BUFFER
IS OUTPUT BUFFER FULL ?(END OF-LINE CONDITION)
NO
YES
PRINT A LINE OF F/C's (LIGHT)
ANY MORE F/C's LEFT?
YES
NO
MOVE A BLANK OVER FINAL COMMA
PRINT FILE/ CATEGORY NUMBERS (LIGHT)
DOUBLE SPACE PRINTER
42

SELECTIVE LIST SUBROUTINE


42
A
BRANCH TO NEXT TERM
END OF DATA?
YES
NO
EXTERNAL SWITCH DPSW2 SET 'ALL' X'FF'
YES
NO
ALL TERMS REQUESTED FOUND, DPSW2=INTERNAL SWITCH?
YES
NO
PRINT TERMS REQUESTED BUT NOT FOUND AND 'NONE ENCOUNTERED' (LIGHT)
RELOAD REGISTERS AND REINITIALIZE SWITCHES
RETURN
IS TERM REQUESTED?
NO
YES
IS TERM 'USE' TYPE, X'01'?
NO
YES
SET INTERNAL SWITCH, USE AND RELATED TYPE BITS, X'03'
PRINT 'USE' TITLE (DARK)
IS TERM 'RELATED' TYPE X'02'?
NO
YES
HAS INTERNAL SWITCH BEEN SET, X'02'?
YES
NO
SET INTERNAL SWITCH, X'03', AND COUNT TOTAL RELATED TERMS IN DATA
PRINT RELATED TERM TITLE AND COUNT
IS TERM 'SEE ALSO' TYPE X'04'?
NO
YES
HAS INTERNAL SWITCH BEEN SET, X'04?
YES
NO
SET INTERNAL SWITCH, X'04'
PRINT SEE ALSO TITLE (DARK)
IS TERM 'SCOPE NOTES' X'08'?
NO
A
YES
HAS INTERNAL SWITCH BEEN SET X'08'?
NO
YES
SET INTERNAL SWITCH, X'08
PRINT SCOPE NOTES TITLE (LIGHT)
CONVERT LINE SEQ. NO. TO DECIMAL
MOVE LINE SEQUENCE NO. TO OUTPUT BUFFER
MOVE TERM TO OUTPUT BUFFER
IS TERM TYPE RELATED?
YES
NO
ANY F/C CODES?
YES
NO
SBCVD
CONVERT F/C TO DECIMAL
MOVE F/C CODES TO OUTPUT AREA
ANY MORE TERMS?
NO
YES
PRINT OUTPUT BUFFER (LIGHT)
A
IS NEXT TERM SAME TYPE?
NO
YES
ANY ROOM ON LINE FOR ENTIRE NEXT TERM?
NO
YES
NEXT TERM SCOPE NOTES TYPE, X'08'?
NO
YES

SELECTIVE LIST SUBROUTINE

CONVERT TO DECIMAL SUBROUTINE


ENTRY
CONVERT NUMBER TO DECIMAL
CHECK LEFT-MOST DIGIT
IS DIGIT=0?
NO
YES
MAKE REMAINING DIGITS AVAILABLE TO MAIN ROUTINE
DISREGARD AND CHECK NEXT DIGIT
IS DIGIT LAST DIGIT?
NO
YES
RETURN

SYNONYM BUILDING PROGRAM (SYNTAPE) PROCESSING LOGIC

The files are opened and a record is read. The contents position is tested for the presence of a Use term. If one exists the record is scanned and the Use term is extracted. It is written out along with the main term with which it is associated. If no Use term is present, the record is ignored. Then the next record is read and processed. As soon as an end-of-file is encountered on the dictionary tape, processing terminates.

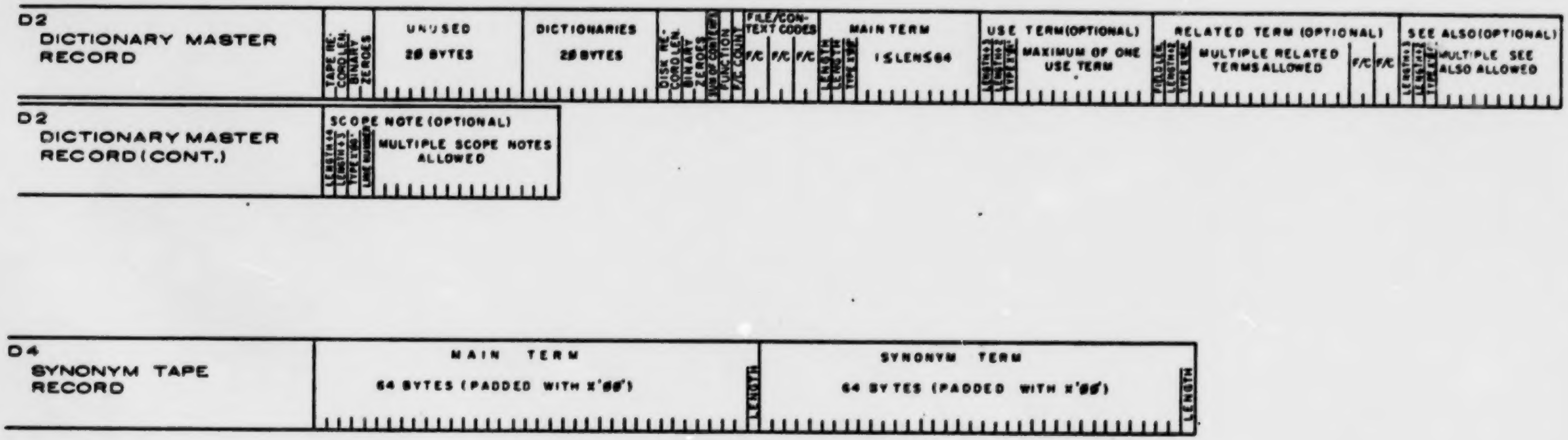
Input

The tape containing the current dictionary is mounted on SYS009. Its format is shown below. For a more detailed description of the Dictionary master record please refer to Appendix A.

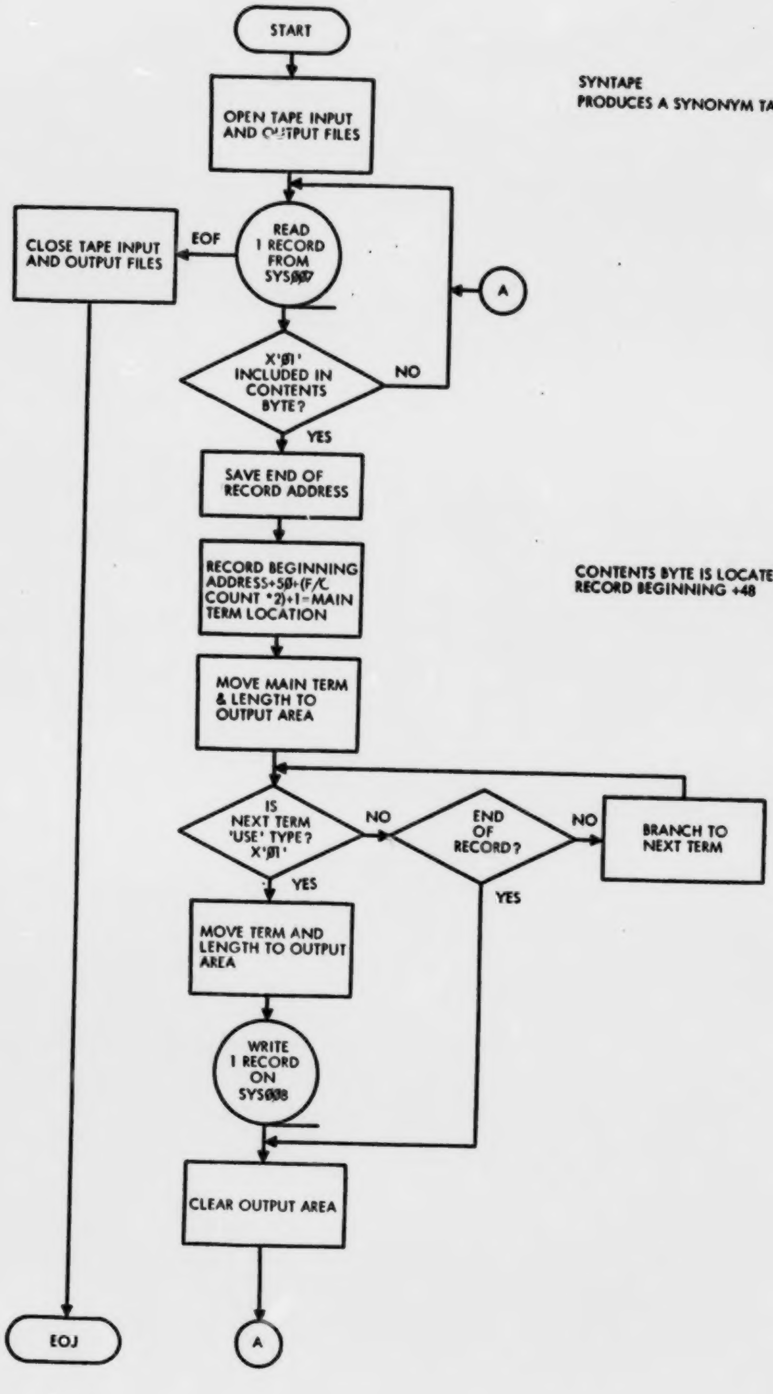
Output

The records containing synonyms and main terms (D4 below) are written on the tape mounted on SYS008. After sorting they will provide synonym input to DPRINT4.

SYNONUM BUILDING PROGRAM (SYNTAPE) RECORD FORMATS


D2
DICTIONARY MASTER RECORD
TAPE RE-CORDLEN.
BINARY ZEROES
UNUSED
20 BYTES
DICTIONARIES
20 BYTES
DISK RE-CORDLEN.
BINARY ZEROES
FUNCTION
F/C COUNT
FILE/CONTEXT CODES
F/C
F/C
F/C
MAIN TERM
LENGTH
LENGTH
1≤LEN≤64
USE TERM(OPTIONAL)
MAXIMUM OF ONE USE TERM
RELATED TERM (OPTIONAL)
MULTIPLE RELATED TERMS ALLOWED
F/C
F/C
SEE ALSO(OPTIONAL)
MULTIPLE SEE ALSO ALLOWED
D2
DICTIONARY MASTER RECORD(CONT.)
SCOPE NOTE (OPTIONAL)
MULTIPLE SCOPE NOTES ALLOWED
D4
SYNONYM TAPE RECORD
MAIN TERM
64 BYTES (PADDED WITH X'00')
LENGTH
SYNONYM TERM
64 BYTES (PADDED WITH X'00')
LENGTH

SYNTAPE
PRODUCES A SYNONYM TAPE FOR DPRINT 4


START
OPEN TAPE INPUT AND OUTPUT FILES
READ 1 RECORD FROM SYS007
EOF
CLOSE TAPE INPUT AND OUTPUT FILES
A
X'01' INCLUDED IN CONTENTS BYTE?
NO
YES
SAVE END OF RECORD ADDRESS
RECORD BEGINNING ADDRESS+50+(F/C COUNT *2)+1=MAIN TERM LOCATION
MOVE MAIN TERM & LENGTH TO OUTPUT AREA
IS NEXT TERM 'USE' TYPE? X'01'
NO
END OF RECORD?
NO
BRANCH TO NEXT TERM
YES
YES
MOVE TERM AND LENGTH TO OUTPUT AREA
WRITE 1 RECORD ON SYS008
CLEAR OUTPUT AREA
EOJ
A

CONTENTS BYTE IS LOCATED AT
RECORD BEGINNING +48

DICTIONARY PRINT PROGRAM (DPRINT4) PROCESSING LOGIC

DPRINT4 Main Control

After initialization, control is transferred to the routine Card which reads a control card and edits it. Then control is returned to this routine.

If a card is in error, a message is printed, and the card is ignored. Cards are read until a system end-of-file card is encountered.

If there are control cards missing, an error message is printed and the job is terminated.

If all control cards are found, the first synonym tape record is read.

A dictionary record is read. If it is an end of file record, control is transferred to subroutine SBRFORM which prints the contents of the print buffer and returns to this program. Any erroneous synonym tape records are printed, the files are closed and the job is terminated.

If the dictionary record is not an end of file record, but this record is not included in the selected dictionaries, this record is not to be printed. If the dictionary record main term and the synonym record main term are equal, synonym records are read until they are unequal. Then the program reads the next dictionary record.

If the dictionary record is not an end of file and this record is included in the list of selected dictionaries, control is transferred to routine SBRFORM which formats the dictionary print record and prints the buffer when full. When control is returned, this program reads the next dictionary record.

Card - Read and Edit Control Cards

This routine reads one card. If the first character encountered is a "C", the card is assumed to be a copy control card; if it is a "D", the card is assumed to be a dump control card; if it is an "L", the card is assumed to be a list control card. If the first character is an "H", the card is a header card and the data is moved to the title positions in the print routine. An "R" indicates a restart card and DPRINT4 positions both tapes to the term specified by the TERM= parameter in the restart card. If the first character is not any of the above, the card is printed along with an error message, and the program reads the next card. If the type of card read has already been encountered, the card is ignored and the next card is read.

If the card is a copy control card, the program scans the card until a slash is encountered. If the next character encountered is a "W", a working copy will be printed. If it is a "P", a "publish" copy will be printed. If neither of the above is encountered or if the end of the card is sense, the card is printed along with an error message. The program reads the next card.

If the copy card is valid, parameters are set up for printing and a switch is set indicating that a valid copy card has been read. The card is printed, then the program reads the next card.

If the control card is a dump control card, the program scans the card until a slash is encountered. The next character encountered is assumed to begin a dump parameter. This program transfers to routine PTEST which tests for the following:

Parameter	Description
RECORD or	Entire Record
AN	Authority of Nomenclature
DICT	Dictionaries
FN	Function
FC	File Context Numbers
SCOPE	Scope Notes
SA	See Also References
RT	Related Terms
USE	Use Terms

If the parameter is valid, the appropriate switch is set. Control is returned, and the program scans to a comma or end of card. If a comma is encountered, the program expects another parameter. If end of card, a switch is set indicating that valid dump card has been read. The card is printed, then the program reads another card. If an error is found or end of card is sensed before it is expected, the card is printed with an error message. The program then reads another card.

If the control card is a list control card, the program scans the card until a slash is encountered. The next character encountered is assumed to begin a dictionary mnemonic. If it is "all" the switch is set to indicate that all records are to be printed. If not, a dictionary equivalence table, DCTTAB, is loaded into core. The mnemonic is converted to its binary equivalent using this table. If the mnemonic is not in the table, the card is printed with an error message. The program reads another card.

A valid mnemonic is entered in a binary list in core. The program then scans the card for a comma or the end of card. If a comma is found, the program expects another dictionary mnemonic to follow. If the end of card is sensed,a switch is set indicating that a valid list card has been read. The card is printed, and the program reads another card.

If an error is found or end of card is sensed before it is expected, the card is printed with an error message. The program reads another card.

SBRFORM Modified for DPRINT4

SBRFORM requires a data address in register 10 and return address in register 14. The data manipulation involves reformatting the dictionary record data and testing for the inclusion of synonyms with the subordinate data for each main term. All binary numbers are converted to their decimal equivalents prior to printing.

The printing is done concurrently with the data manipulation. That is, as soon as a data buffer is full, a test is made to determine whether or not the printer is busy (i.e., has not yet completed the last print operation). If it is, SBRFORM waits until it has finished. As soon as the printer becomes available, an instruction is executed which causes the printer to begin printing the buffer just filled and the data manipulation is resumed. At the conclusion of DPRINT4, the remaining data in the buffer last filled is printed.

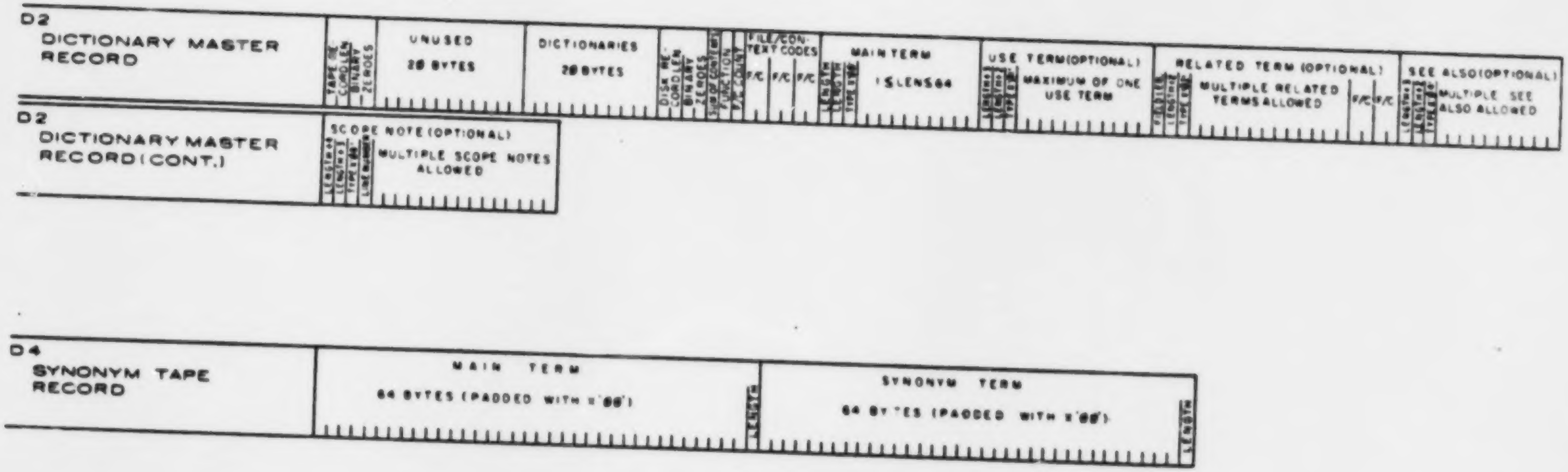
Input

Input to the program consists of control cards as discussed in the processing logic section and the tape containing the current dictionary. The tape record format is labelled D2 below. For a more detailed discussion of the record format refer to Appendix A.

Output

The listing of the dictionary appears on the printer which must be assigned to SYS021. A carriage control tape which contains only channel 1 punches must be used.

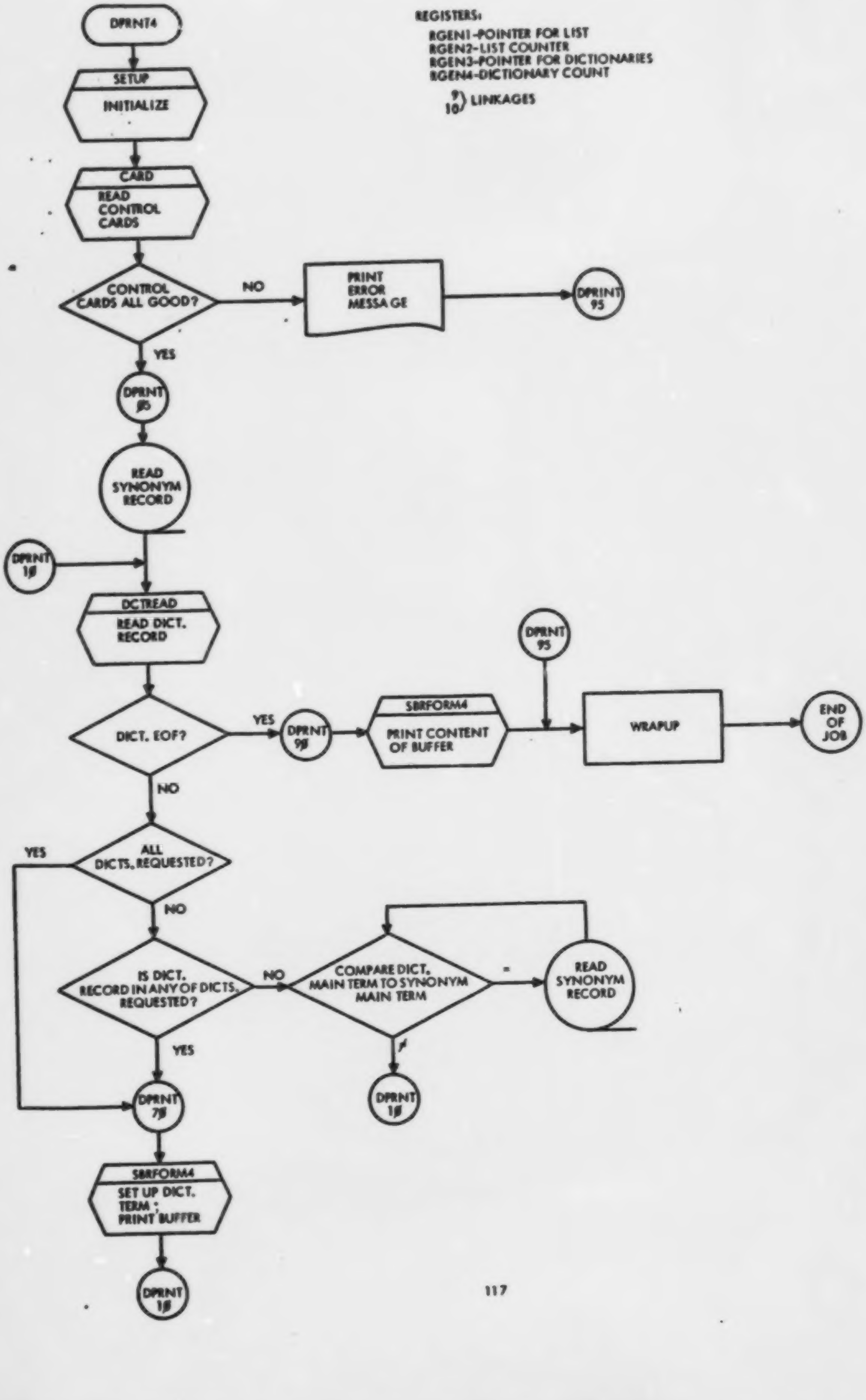
DICTIONARY PRINT PROGRAM (DPRINT4) RECORD FORMATS

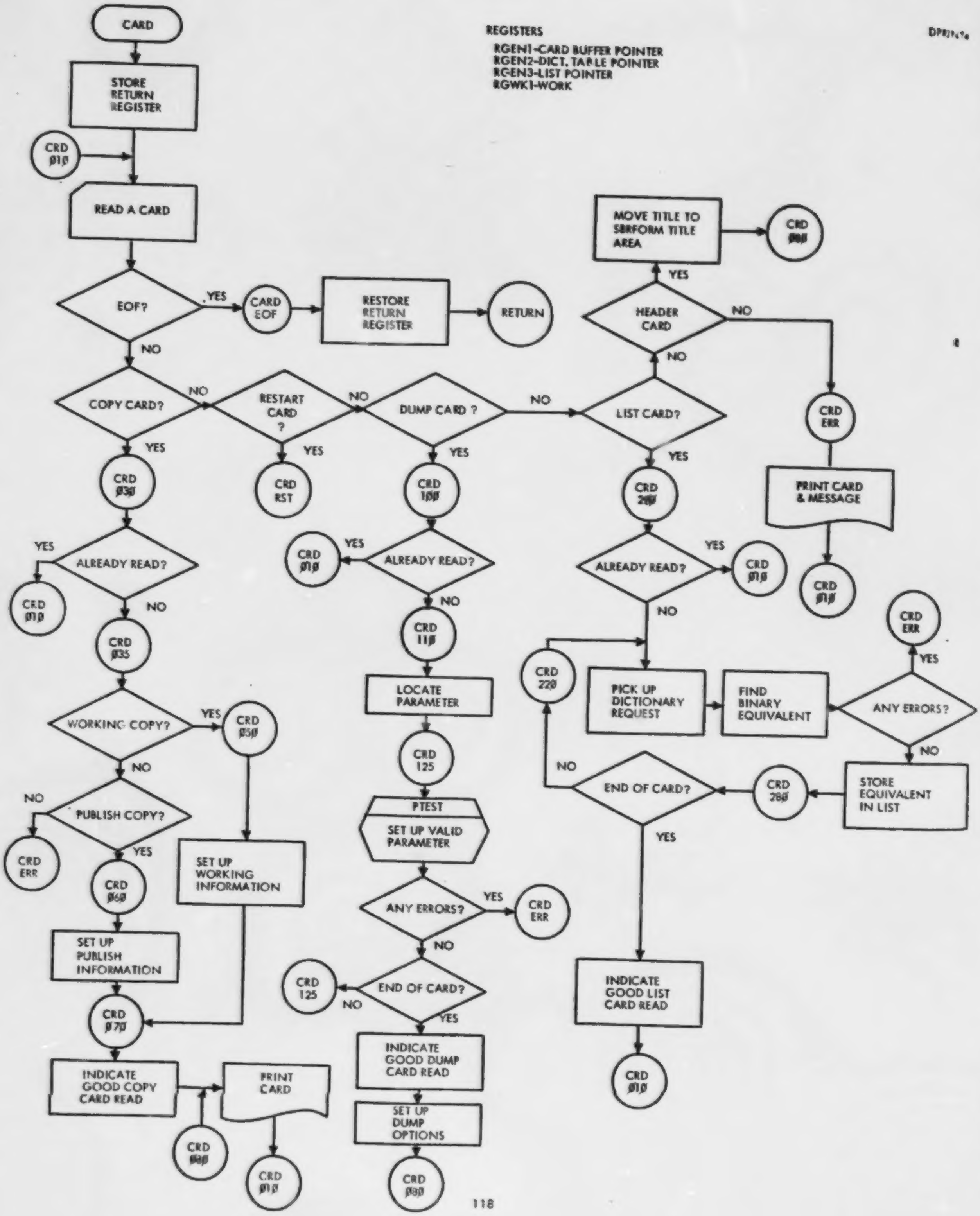

D2
DICTIONARY MASTER RECORD
TAPE RE-CORDLEN
BINARY ZEROES
UNUSED
20 BYTES
DICTIONARIES
20 BYTES
DISK RE-CORDLEN
BINARY ZEROES
FUNCTION
F/C COUNT
FILE/CON-TEXT CODES
F/C
F/C
F/C
MAIN TERM
1≤LEN≤64
USE TERM(OPTIONAL)
MAXIMUM OF ONE USE TERM
RELATED TERM (OPTIONAL)
MULTIPLE RELATED TERMS ALLOWED
F/C
F/C
SEE ALSO(OPTIONAL)
MULTIPLE SEE ALSO ALLOWED
D2
DICTIONARY MASTER RECORD(CONT.)
SCOPE NOTE (OPTIONAL)
MULTIPLE SCOPE NOTES ALLOWED
D4
SYNONYM TAPE RECORD
MAIN TERM
64 BYTES (PADDED WITH X'00')
LENGTH
SYNONYM TERM
64 BYTES (PADDED WITH X'00')
LENGTH

REGISTERS:

RGEN1-POINTER FOR LIST
RGEN2-LIST COUNTER
RGEN3-POINTER FOR DICTIONARIES
RGEN4-DICTIONARY COUNT

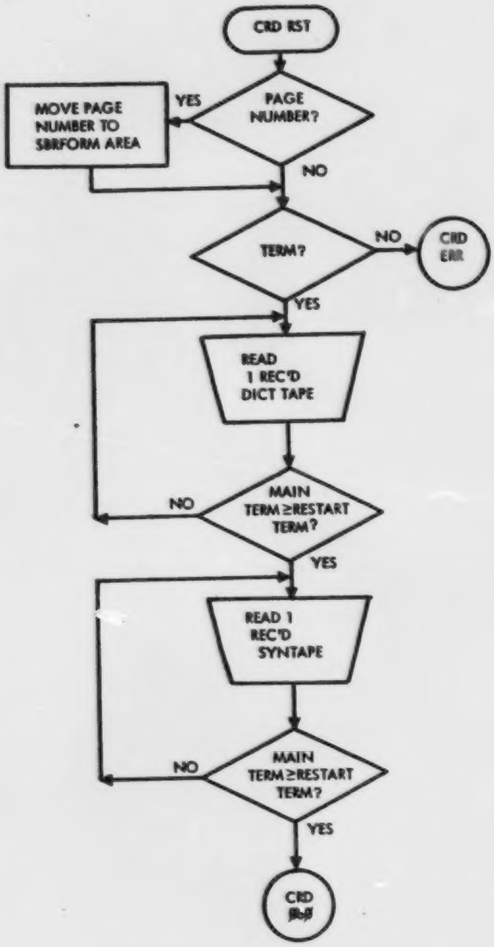
9
10 } LINKAGES

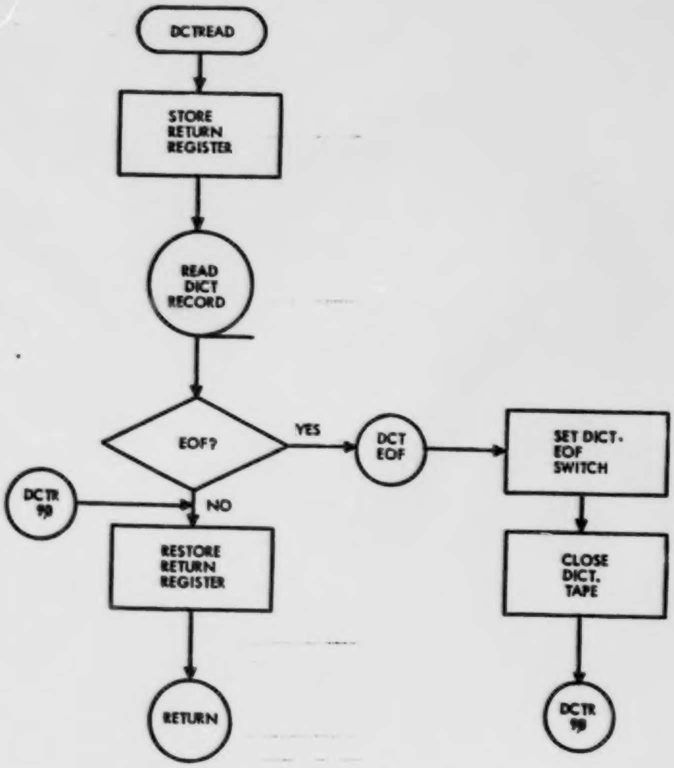

DPRNT4
SETUP
INITIALIZE
CARD
READ CONTROL CARDS
CONTROL CARDS ALL GOOD?
NO
PRINT ERROR MESSAGE
DPRINT 95
YES
DPRNT 05
READ SYNONYM RECORD
DPRNT 10
DCTREAD
READ DICT. RECORD
DPRNT 95
DICT. EOF?
YES
DPRNT 90
SBRFORM4
PRINT CONTENT OF BUFFER
WRAPUP
END OF JOB
NO
ALL DICTS. REQUESTED?
YES
NO
IS DICT. RECORD IN ANY OF DICTS. REQUESTED?
NO
COMPARE DICT. MAIN TERM TO SYNONYM MAIN TERM
=
READ SYNONYM RECORD
≠
DPRNT 10
YES
DPRNT 70
SBRFORM4
SET UP DICT. TERM ; PRINT BUFFER
DPRNT 10

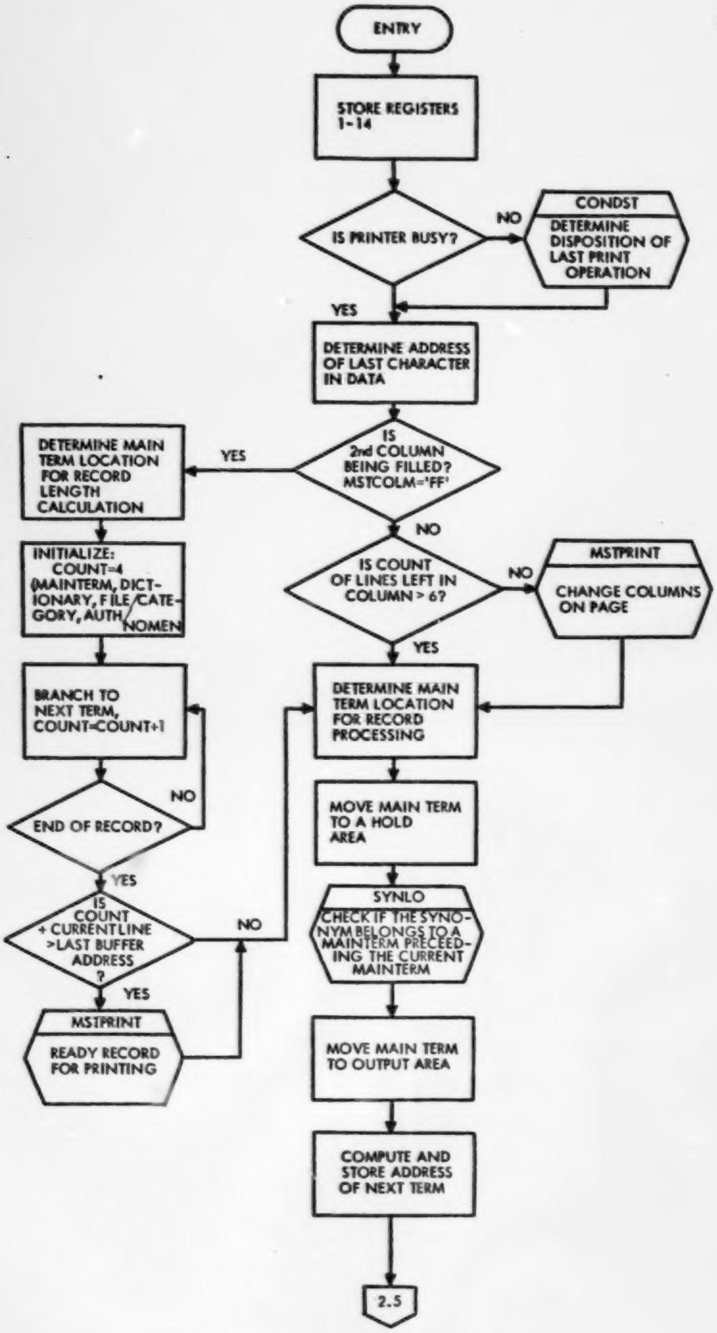

CARD
STORE RETURN REGISTER
CRD 010
READ A CARD
EOF?
YES
CARD EOF
RESTORE RETURN REGISTER
RETURN
NO
COPY CARD?
NO
RESTART CARD ?
NO
DUMP CARD ?
NO
LIST CARD?
YES
CRD 030
ALREADY READ?
YES
CRD 010
NO
CRD 035
WORKING COPY?
YES
CRD 050
NO
PUBLISH COPY?
NO
CRD ERR
YES
CRD 060
SET UP PUBLISH INFORMATION
SET UP WORKING INFORMATION
CRD 070
INDICATE GOOD COPY CARD READ
PRINT CARD
CRD 080
CRD 010
YES
CRD RST
YES
CRD 100
ALREADY READ?
YES
CRD 010
NO
CRD 110
LOCATE PARAMETER
CRD 125
PTEST
SET UP VALID PARAMETER
ANY ERRORS?
YES
CRD ERR
NO
END OF CARD?
NO
CRD 125
YES
INDICATE GOOD DUMP CARD READ
SET UP DUMP OPTIONS
CRD 080
MOVE TITLE TO SBRFORM TITLE AREA
CRD 080
YES
HEADER CARD
NO
NO
CRD ERR
PRINT CARD & MESSAGE
CRD 010
YES
CRD 200
ALREADY READ?
YES
CRD 010
NO
CRD 220
PICK UP DICTIONARY REQUEST
FIND BINARY EQUIVALENT
ANY ERRORS?
YES
CRD ERR
NO
STORE EQUIVALENT IN LIST
CRD 280
END OF CARD?
NO
YES
INDICATE GOOD LIST CARD READ
CRD 010

REGISTERS

RGEN1-CARD BUFFER POINTER
RGEN2-DICT. TABLE POINTER
RGEN3-LIST POINTER
RGWK1-WORK

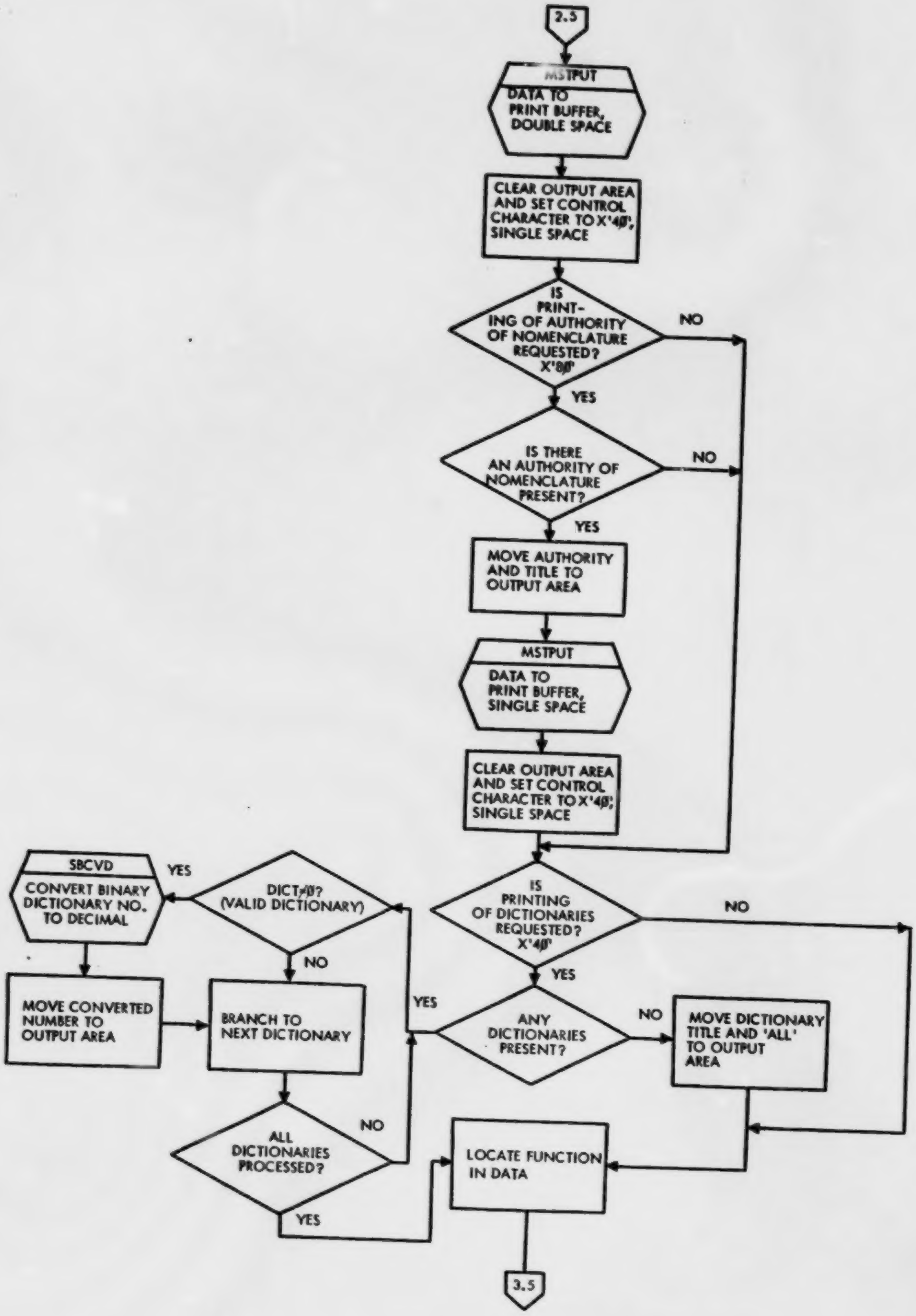

CRD RST
PAGE NUMBER?
YES
MOVE PAGE NUMBER TO SBRFORM AREA
NO
TERM?
NO
CRD ERR
YES
READ 1 REC'D DICT TAPE
MAIN TERM ≥ RESTART TERM?
NO
YES
READ 1 REC'D SYNTAPE
MAIN TERM ≥ RESTART TERM?
NO
YES
CRD

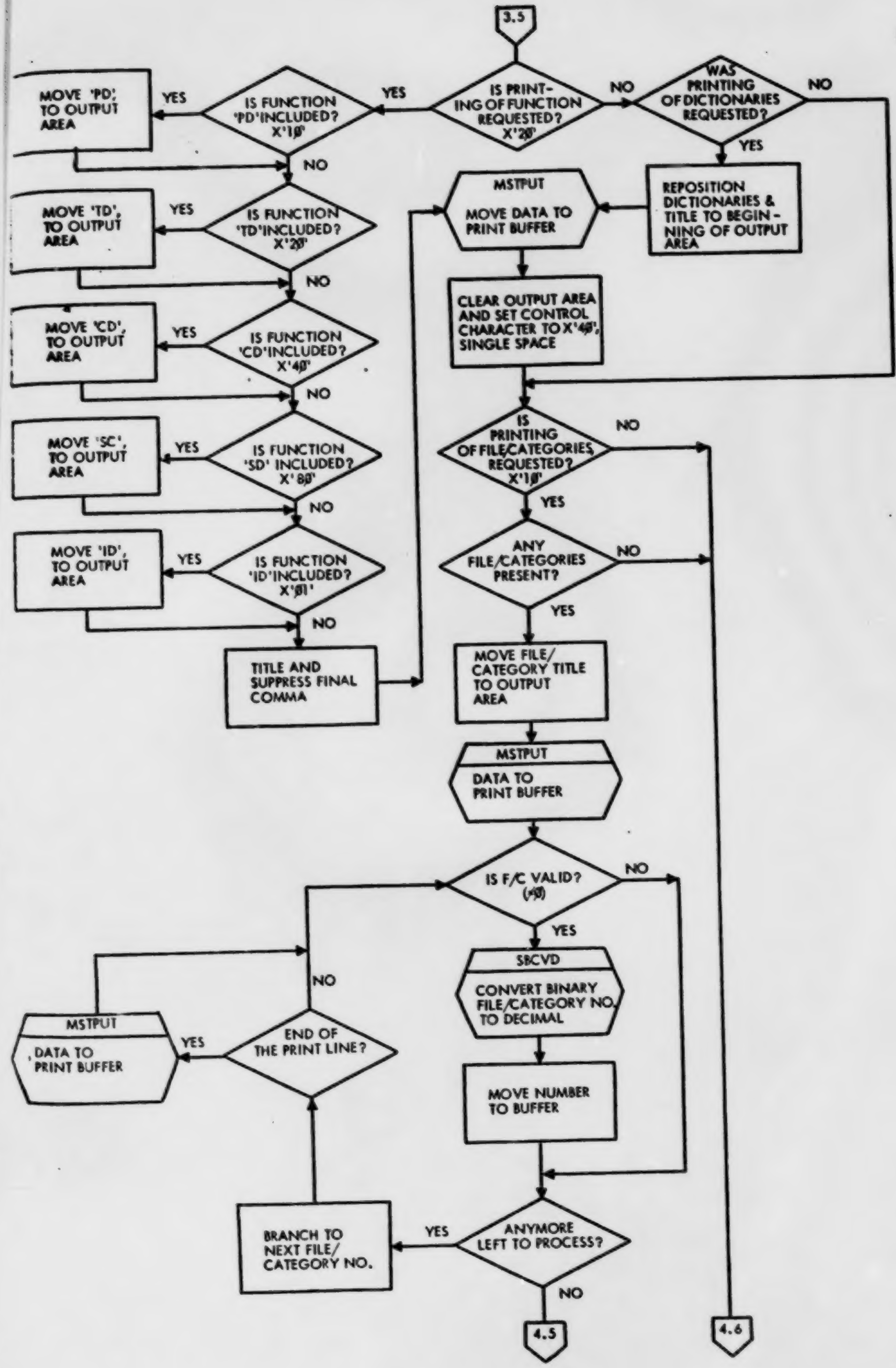

DCTREAD
STORE RETURN REGISTER
READ DICT RECORD
EOF?
YES
NO
DCT EOF
SET DICT. EOF SWITCH
CLOSE DICT. TAPE
DCTR 99
RESTORE RETURN REGISTER
RETURN
DCTR 99


ENTRY
STORE REGISTERS 1-14
IS PRINTER BUSY?
NO
CONDST
DETERMINE DISPOSITION OF LAST PRINT OPERATION
YES
DETERMINE ADDRESS OF LAST CHARACTER IN DATA
IS 2nd COLUMN BEING FILLED? MSTCOLM='FF'
YES
DETERMINE MAIN TERM LOCATION FOR RECORD LENGTH CALCULATION
INITIALIZE: COUNT=4 (MAINTERM, DICTIONARY, FILE/CATEGORY, AUTH/NOMEN
BRANCH TO NEXT TERM, COUNT=COUNT+1
END OF RECORD?
NO
YES
IS COUNT + CURRENTLINE >LAST BUFFER ADDRESS ?
NO
YES
MSTPRINT
READY RECORD FOR PRINTING
NO
IS COUNT OF LINES LEFT IN COLUMN > 6?
NO
MSTPRINT
CHANGE COLUMNS ON PAGE
YES
DETERMINE MAIN TERM LOCATION FOR RECORD PROCESSING
MOVE MAIN TERM TO A HOLD AREA
SYNLO
CHECK IF THE SYNONYM BELONGS TO A MAINTERM PRECEEDING THE CURRENT MAINTERM
MOVE MAIN TERM TO OUTPUT AREA
COMPUTE AND STORE ADDRESS OF NEXT TERM
2.5

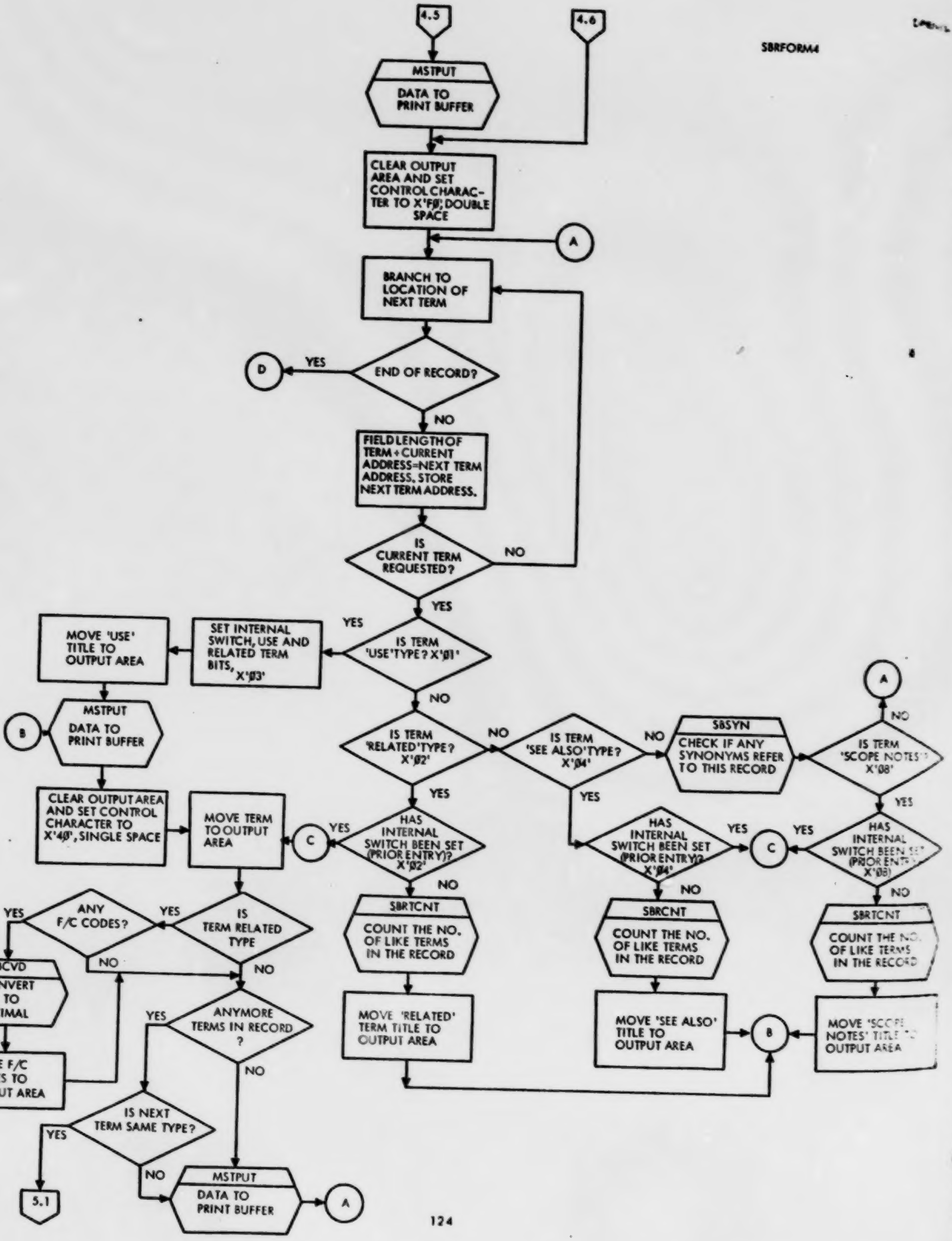
DPRINT4

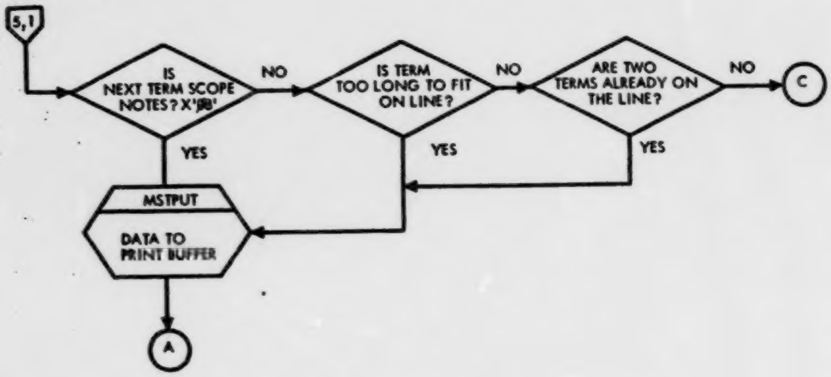
SBRFORM4

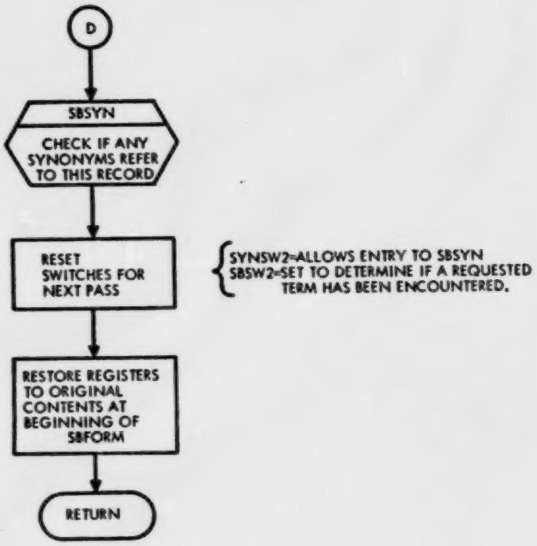

2.5
MSTPUT
DATA TO PRINT BUFFER, DOUBLE SPACE
CLEAR OUTPUT AREA AND SET CONTROL CHARACTER TO X'40', SINGLE SPACE
IS PRINTING OF AUTHORITY OF NOMENCLATURE REQUESTED? X'80'
NO
YES
IS THERE AN AUTHORITY OF NOMENCLATURE PRESENT?
NO
YES
MOVE AUTHORITY AND TITLE TO OUTPUT AREA
MSTPUT
DATA TO PRINT BUFFER, SINGLE SPACE
CLEAR OUTPUT AREA AND SET CONTROL CHARACTER TO X'40', SINGLE SPACE
IS PRINTING OF DICTIONARIES REQUESTED? X'40'
NO
YES
ANY DICTIONARIES PRESENT?
NO
MOVE DICTIONARY TITLE AND 'ALL' TO OUTPUT AREA
YES
DICT≠0? (VALID DICTIONARY)
YES
SBCVD
CONVERT BINARY DICTIONARY NO. TO DECIMAL
MOVE CONVERTED NUMBER TO OUTPUT AREA
NO
BRANCH TO NEXT DICTIONARY
ALL DICTIONARIES PROCESSED?
NO
YES
LOCATE FUNCTION IN DATA
3.5


3.5
IS PRINTING OF FUNCTION REQUESTED? X'20'
YES
NO
WAS PRINTING OF DICTIONARIES REQUESTED?
NO
YES
IS FUNCTION 'PD' INCLUDED? X'10'
YES
MOVE 'PD', TO OUTPUT AREA
NO
IS FUNCTION 'TD' INCLUDED? X'20'
YES
MOVE 'TD', TO OUTPUT AREA
NO
IS FUNCTION 'CD' INCLUDED? X'40'
YES
MOVE 'CD', TO OUTPUT AREA
NO
IS FUNCTION 'SD' INCLUDED? X'80'
YES
MOVE 'SC', TO OUTPUT AREA
NO
IS FUNCTION 'ID' INCLUDED? X'01'
YES
MOVE 'ID', TO OUTPUT AREA
NO
TITLE AND SUPPRESS FINAL COMMA
REPOSITION DICTIONARIES & TITLE TO BEGINNING OF OUTPUT AREA
MSTPUT
MOVE DATA TO PRINT BUFFER
CLEAR OUTPUT AREA AND SET CONTROL CHARACTER TO X'40', SINGLE SPACE
IS PRINTING OF FILE/CATEGORIES REQUESTED? X'10'
NO
YES
ANY FILE/CATEGORIES PRESENT?
NO
YES
MOVE FILE/ CATEGORY TITLE TO OUTPUT AREA
MSTPUT
DATA TO PRINT BUFFER
IS F/C VALID? (≠0)
NO
YES
SBCVD
CONVERT BINARY FILE/CATEGORY NO. TO DECIMAL
MOVE NUMBER TO BUFFER
ANYMORE LEFT TO PROCESS?
YES
NO
BRANCH TO NEXT FILE/ CATEGORY NO.
END OF THE PRINT LINE?
NO
YES
MSTPUT
, DATA TO PRINT BUFFER
4.5
4.6

SBRFORM4

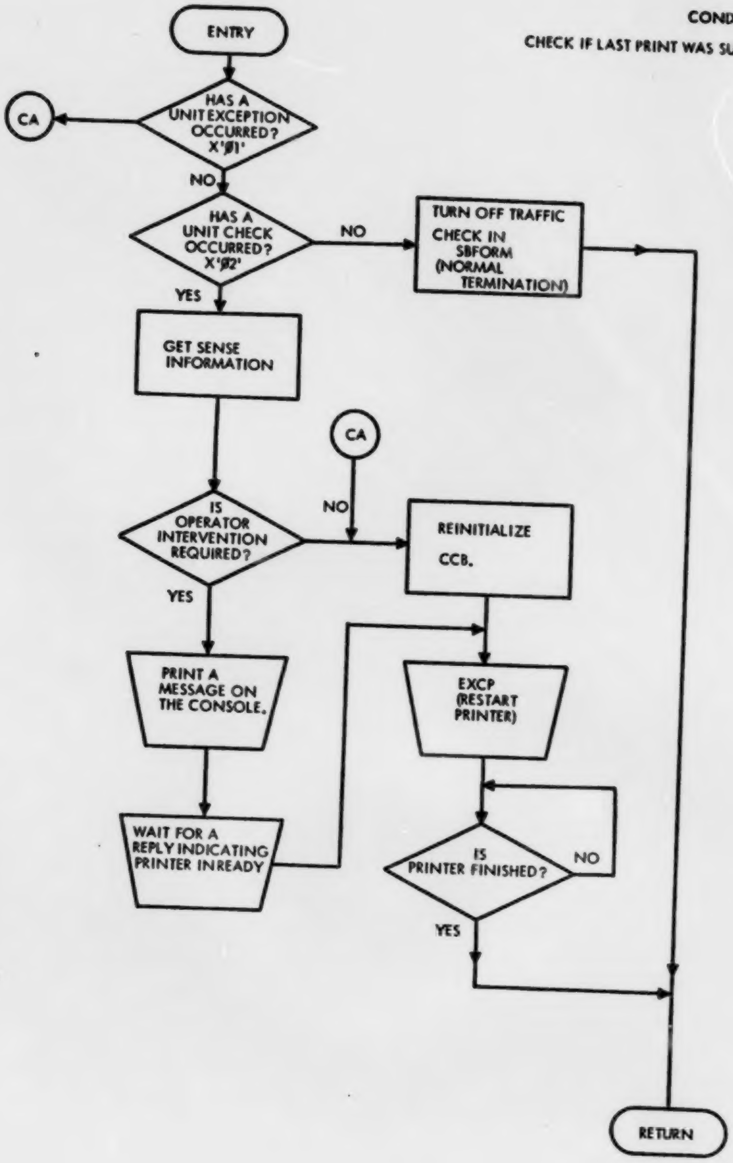

4.5
4.6
MSTPUT
DATA TO PRINT BUFFER
CLEAR OUTPUT AREA AND SET CONTROL CHARACTER TO X'F0', DOUBLE SPACE
A
BRANCH TO LOCATION OF NEXT TERM
D
YES
END OF RECORD?
NO
FIELD LENGTH OF TERM + CURRENT ADDRESS = NEXT TERM ADDRESS. STORE NEXT TERM ADDRESS.
IS CURRENT TERM REQUESTED?
NO
YES
MOVE 'USE' TITLE TO OUTPUT AREA
SET INTERNAL SWITCH, USE AND RELATED TERM BITS, X'03'
YES
IS TERM 'USE' TYPE? X'01'
NO
A
NO
MSTPUT
B
DATA TO PRINT BUFFER
IS TERM 'RELATED' TYPE? X'02'
NO
IS TERM 'SEE ALSO' TYPE? X'04'
NO
SBSYN
CHECK IF ANY SYNONYMS REFER TO THIS RECORD
IS TERM 'SCOPE NOTES'? X'08'
YES
YES
YES
CLEAR OUTPUT AREA AND SET CONTROL CHARACTER TO X'40', SINGLE SPACE
MOVE TERM TO OUTPUT AREA
C
YES
HAS INTERNAL SWITCH BEEN SET (PRIOR ENTRY)? X'02'
HAS INTERNAL SWITCH BEEN SET (PRIOR ENTRY)? X'04'
YES
C
YES
HAS INTERNAL SWITCH BEEN SET (PRIOR ENTRY)? X'08'
NO
NO
NO
YES
ANY F/C CODES?
YES
IS TERM RELATED TYPE
SBRTCNT
COUNT THE NO. OF LIKE TERMS IN THE RECORD
SBRCNT
COUNT THE NO. OF LIKE TERMS IN THE RECORD
SBRTCNT
COUNT THE NO. OF LIKE TERMS IN THE RECORD
NO
NO
SBCVD
CONVERT F/C TO DECIMAL
YES
ANYMORE TERMS IN RECORD ?
MOVE 'RELATED' TERM TITLE TO OUTPUT AREA
MOVE 'SEE ALSO' TITLE TO OUTPUT AREA
B
MOVE 'SCOPE NOTES' TITLE TO OUTPUT AREA
MOVE F/C CODES TO OUTPUT AREA
NO
IS NEXT TERM SAME TYPE?
YES
NO
MSTPUT
5.1
DATA TO PRINT BUFFER
A


5,1
IS NEXT TERM SCOPE NOTES? X'08'
NO
IS TERM TOO LONG TO FIT ON LINE?
NO
ARE TWO TERMS ALREADY ON THE LINE?
NO
C
YES
YES
YES
MSTPUT
DATA TO PRINT BUFFER
A

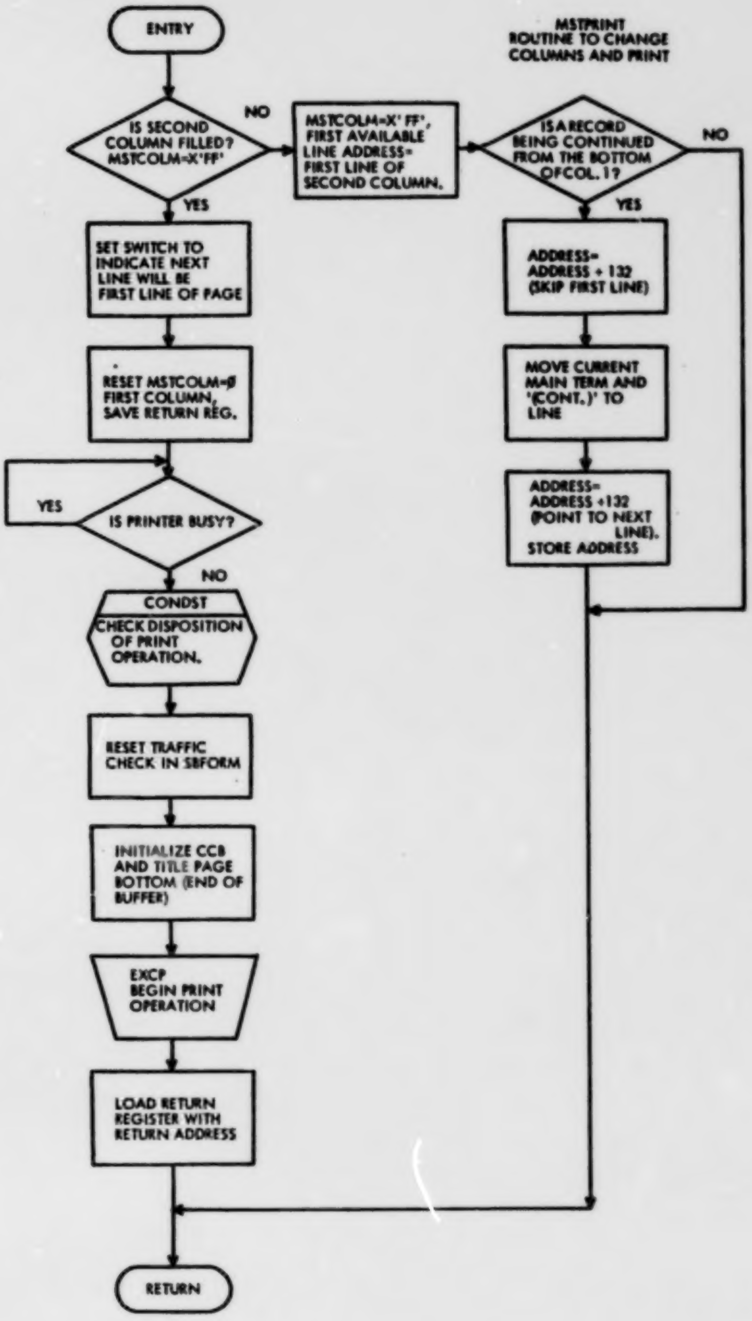

D
SBSYN
CHECK IF ANY SYNONYMS REFER TO THIS RECORD
RESET SWITCHES FOR NEXT PASS
SYNSW2=ALLOWS ENTRY TO SBSYN
SBSW2=SET TO DETERMINE IF A REQUESTED TERM HAS BEEN ENCOUNTERED.
RESTORE REGISTERS TO ORIGINAL CONTENTS AT BEGINNING OF SBFORM
RETURN

CONDST

CHECK IF LAST PRINT WAS SUCCESSFULLY COMPLETED

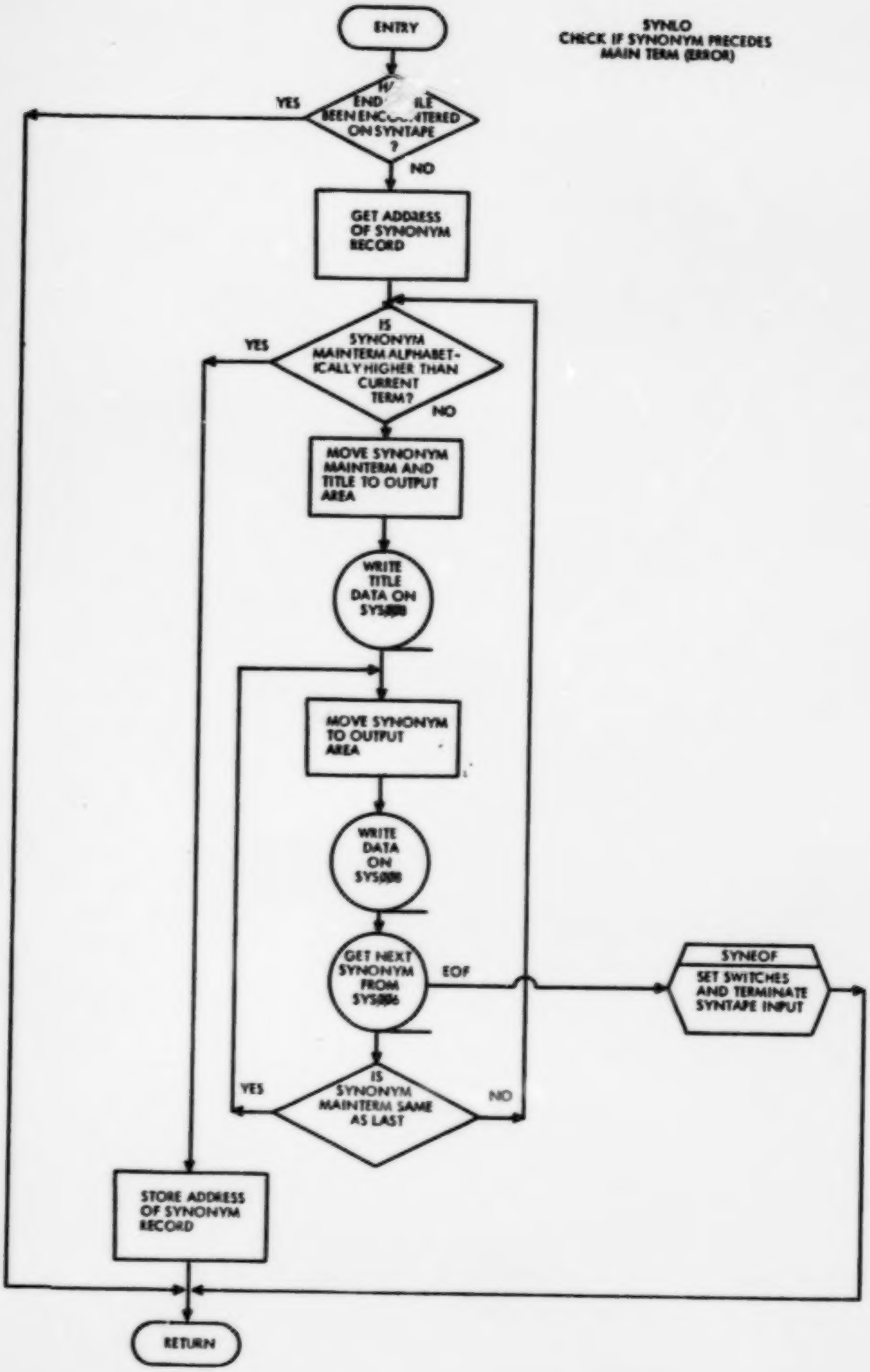

ENTRY
HAS A UNIT EXCEPTION OCCURRED? X'01'
CA
NO
HAS A UNIT CHECK OCCURRED? X'02'
NO
TURN OFF TRAFFIC CHECK IN SBFORM (NORMAL TERMINATION)
YES
GET SENSE INFORMATION
CA
IS OPERATOR INTERVENTION REQUIRED?
NO
REINITIALIZE CCB.
YES
PRINT A MESSAGE ON THE CONSOLE.
EXCP (RESTART PRINTER)
WAIT FOR A REPLY INDICATING PRINTER INREADY
IS PRINTER FINISHED?
NO
YES
RETURN

DPRINT4


MSTPRINT
ROUTINE TO CHANGE COLUMNS AND PRINT
ENTRY
IS SECOND COLUMN FILLED? MSTCOLM=X'FF'
NO
YES
MSTCOLM=X' FF', FIRST AVAILABLE LINE ADDRESS= FIRST LINE OF SECOND COLUMN.
IS A RECORD BEING CONTINUED FROM THE BOTTOM OF COL. 1?
NO
YES
SET SWITCH TO INDICATE NEXT LINE WILL BE FIRST LINE OF PAGE
ADDRESS= ADDRESS + 132 (SKIP FIRST LINE)
RESET MSTCOLM=0 FIRST COLUMN, SAVE RETURN REG.
MOVE CURRENT MAIN TERM AND '(CONT.)' TO LINE
ADDRESS= ADDRESS +132 (POINT TO NEXT LINE). STORE ADDRESS
YES
IS PRINTER BUSY?
NO
CONDST
CHECK DISPOSITION OF PRINT OPERATION.
RESET TRAFFIC CHECK IN SBFORM
INITIALIZE CCB AND TITLE PAGE BOTTOM (END OF BUFFER)
EXCP
BEGIN PRINT OPERATION
LOAD RETURN REGISTER WITH RETURN ADDRESS
RETURN

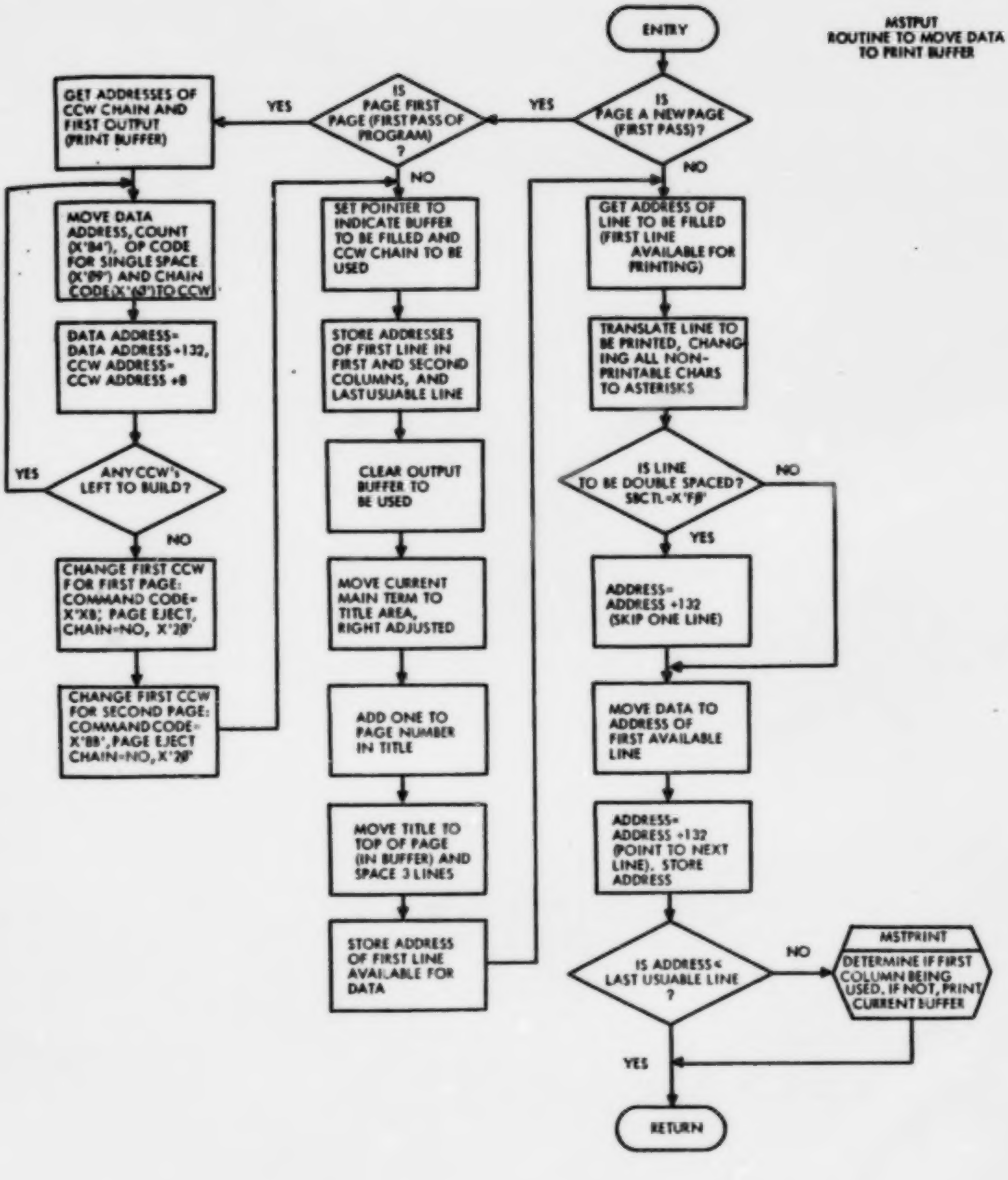
SYNLO
CHECK IF SYNONYM PRECEDES
MAIN TERM (ERROR)

DPMN14


ENTRY
HAS END OF FILE BEEN ENCOUNTERED ON SYNTAPE ?
YES
NO
GET ADDRESS OF SYNONYM RECORD
IS SYNONYM MAINTERM ALPHABETICALLY HIGHER THAN CURRENT TERM?
YES
NO
MOVE SYNONYM MAINTERM AND TITLE TO OUTPUT AREA
WRITE TITLE DATA ON SYS008
MOVE SYNONYM TO OUTPUT AREA
WRITE DATA ON SYS008
GET NEXT SYNONYM FROM SYS006
EOF
SYNEOF
SET SWITCHES AND TERMINATE SYNTAPE INPUT
IS SYNONYM MAINTERM SAME AS LAST
YES
NO
STORE ADDRESS OF SYNONYM RECORD
RETURN

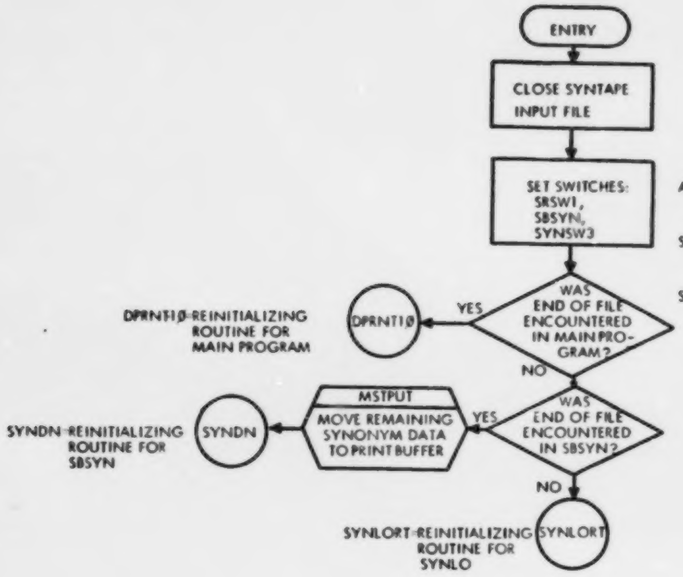
DPRINT4

MSTPUT
ROUTINE TO MOVE DATA TO PRINT BUFFER


ENTRY
IS PAGE A NEW PAGE (FIRST PASS)?
YES
NO
IS PAGE FIRST PAGE (FIRST PASS OF PROGRAM) ?
YES
NO
GET ADDRESSES OF CCW CHAIN AND FIRST OUTPUT (PRINT BUFFER)
MOVE DATA ADDRESS, COUNT (X'84'), OP CODE FOR SINGLE SPACE (X'09') AND CHAIN CODE (X'40') TO CCW
DATA ADDRESS= DATA ADDRESS +132, CCW ADDRESS= CCW ADDRESS +8
ANY CCW's LEFT TO BUILD?
YES
NO
CHANGE FIRST CCW FOR FIRST PAGE: COMMAND CODE= X'XB; PAGE EJECT, CHAIN=NO, X'20'
CHANGE FIRST CCW FOR SECOND PAGE: COMMAND CODE= X'8B', PAGE EJECT CHAIN=NO, X'20'
SET POINTER TO INDICATE BUFFER TO BE FILLED AND CCW CHAIN TO BE USED
STORE ADDRESSES OF FIRST LINE IN FIRST AND SECOND COLUMNS, AND LAST USUABLE LINE
CLEAR OUTPUT BUFFER TO BE USED
MOVE CURRENT MAIN TERM TO TITLE AREA, RIGHT ADJUSTED
ADD ONE TO PAGE NUMBER IN TITLE
MOVE TITLE TO TOP OF PAGE (IN BUFFER) AND SPACE 3 LINES
STORE ADDRESS OF FIRST LINE AVAILABLE FOR DATA
GET ADDRESS OF LINE TO BE FILLED (FIRST LINE AVAILABLE FOR PRINTING)
TRANSLATE LINE TO BE PRINTED, CHANGING ALL NON-PRINTABLE CHARS TO ASTERISKS
IS LINE TO BE DOUBLE SPACED? SBCTL=X'F0'
NO
YES
ADDRESS= ADDRESS +132 (SKIP ONE LINE)
MOVE DATA TO ADDRESS OF FIRST AVAILABLE LINE
ADDRESS= ADDRESS +132 (POINT TO NEXT LINE). STORE ADDRESS
IS ADDRESS < LAST USUABLE LINE ?
NO
MSTPRINT
DETERMINE IF FIRST COLUMN BEING USED. IF NOT, PRINT CURRENT BUFFER
YES
RETURN

SYNEOF

SYNTAPE END OF FILE ROUTINE


ENTRY
CLOSE SYNTAPE INPUT FILE
SET SWITCHES: SRSW1, SBSYN, SYNSW3
WAS END OF FILE ENCOUNTERED IN MAIN PROGRAM?
YES
NO
DPRNT10
DPRNT10-REINITIALIZING ROUTINE FOR MAIN PROGRAM
WAS END OF FILE ENCOUNTERED IN SBSYN?
YES
NO
MSTPUT
MOVE REMAINING SYNONYM DATA TO PRINTBUFFER
SYNDN
SYNDN-REINITIALIZING ROUTINE FOR SBSYN
SYNLORT
SYNLORT-REINITIALIZING ROUTINE FOR SYNLO

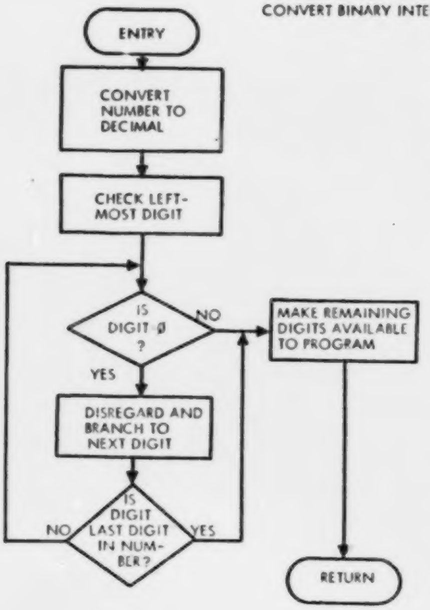
ARSW1-DISALLOWS ENTRY TO MAIN PROGRAM ROUTINE WHICH DETERMINES IF SYNONYMS BELONG TO NON-REQUESTED MAIN TERMS.

SBSYN-DISALLOWS ENTRY TO SBFORM ROUTINE WHICH MATCHES VALID SYNONYMS WITH REQUESTED MAIN TERMS.

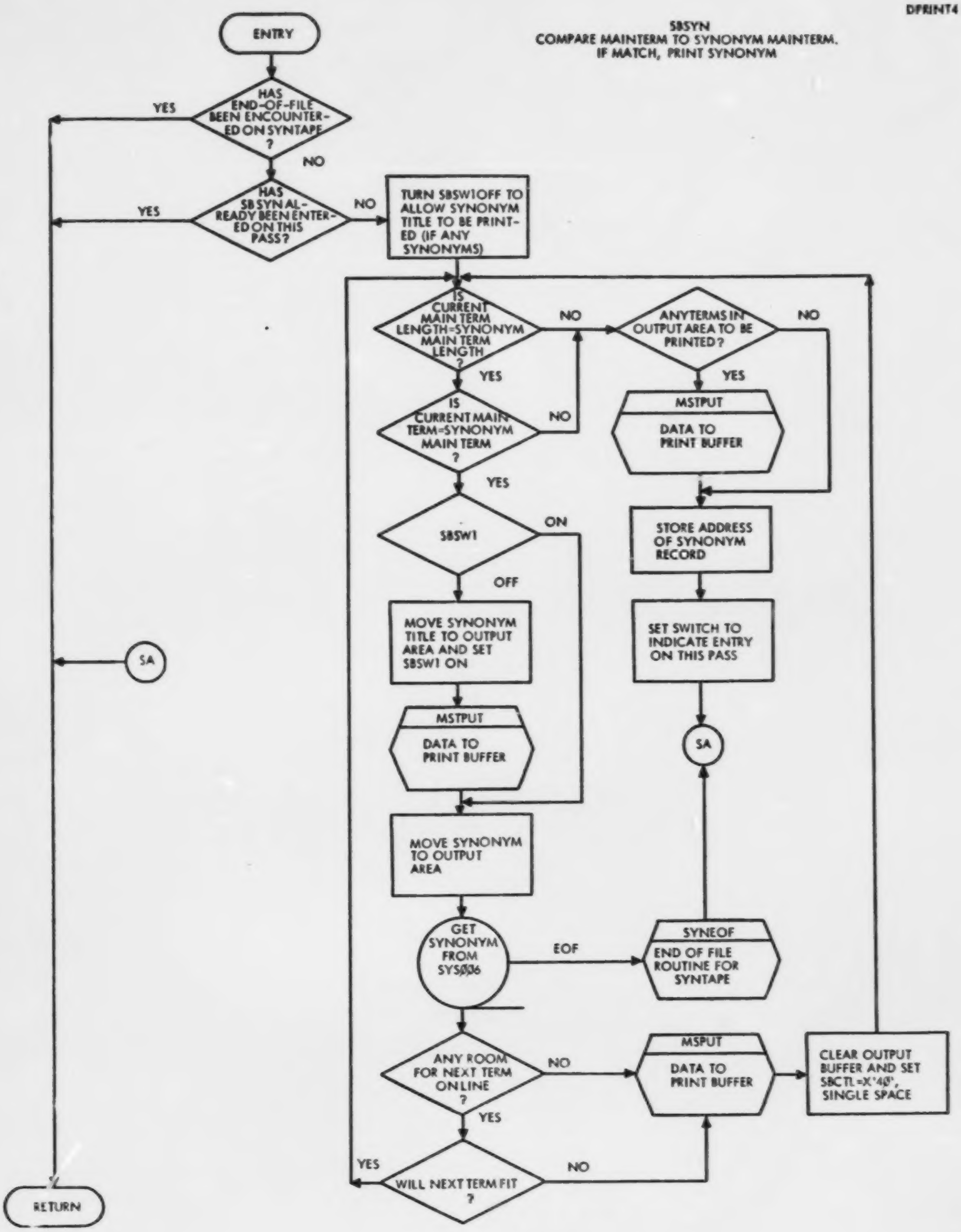
SYNSW3-DISALLOWS ENTRY TO SBFORM ROUTINE WHICH ELIMINATES INVALID SYNONYMS FROM PRINT

SBCVD

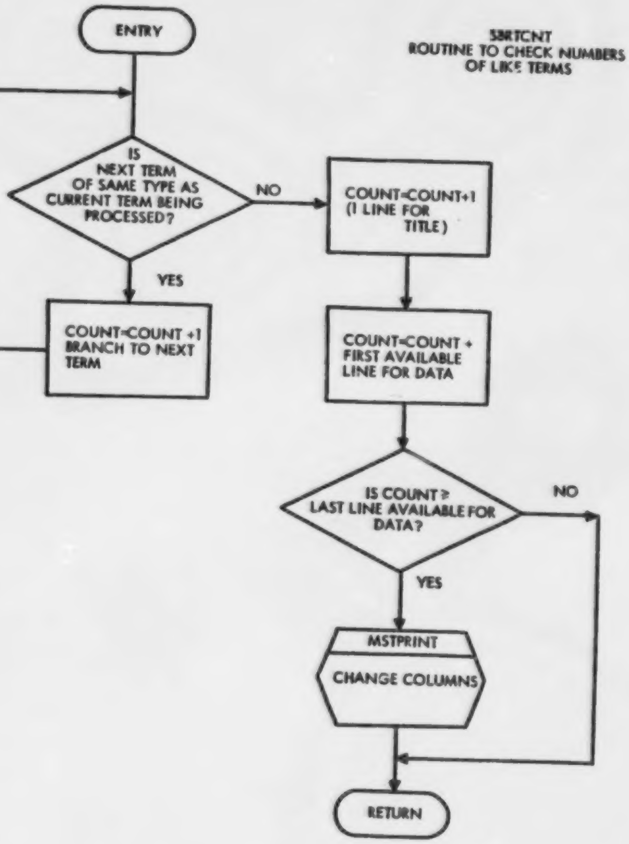
CONVERT BINARY INTEGERS TO DECIMAL


ENTRY
CONVERT NUMBER TO DECIMAL
CHECK LEFT-MOST DIGIT
IS DIGIT 0 ?
NO
YES
MAKE REMAINING DIGITS AVAILABLE TO PROGRAM
DISREGARD AND BRANCH TO NEXT DIGIT
IS DIGIT LAST DIGIT IN NUMBER?
NO
YES
RETURN

SBSYN
COMPARE MAINTERM TO SYNONYM MAINTERM.
IF MATCH, PRINT SYNONYM


ENTRY
HAS END-OF-FILE BEEN ENCOUNTERED ON SYNTAPE ?
YES
NO
HAS SBSYN ALREADY BEEN ENTERED ON THIS PASS?
YES
NO
TURN SBSW1 OFF TO ALLOW SYNONYM TITLE TO BE PRINTED (IF ANY SYNONYMS)
IS CURRENT MAIN TERM LENGTH=SYNONYM MAIN TERM LENGTH ?
NO
YES
ANY TERMS IN OUTPUT AREA TO BE PRINTED?
NO
YES
MSTPUT
DATA TO PRINT BUFFER
IS CURRENT MAIN TERM=SYNONYM MAIN TERM ?
NO
YES
SBSW1
ON
OFF
STORE ADDRESS OF SYNONYM RECORD
MOVE SYNONYM TITLE TO OUTPUT AREA AND SET SBSW1 ON
SET SWITCH TO INDICATE ENTRY ON THIS PASS
SA
SA
MSTPUT
DATA TO PRINT BUFFER
MOVE SYNONYM TO OUTPUT AREA
GET SYNONYM FROM SYS006
EOF
SYNEOF
END OF FILE ROUTINE FOR SYNTAPE
ANY ROOM FOR NEXT TERM ON LINE ?
NO
MSPUT
DATA TO PRINT BUFFER
CLEAR OUTPUT BUFFER AND SET SBCTL=X'40', SINGLE SPACE
YES
YES
WILL NEXT TERM FIT ?
NO
RETURN

SBRTCNT
ROUTINE TO CHECK NUMBERS
OF LIKE TERMS


ENTRY
IS NEXT TERM OF SAME TYPE AS CURRENT TERM BEING PROCESSED?
NO
YES
COUNT=COUNT +1 BRANCH TO NEXT TERM
COUNT=COUNT+1 (1 LINE FOR TITLE)
COUNT=COUNT + FIRST AVAILABLE LINE FOR DATA
IS COUNT ≥ LAST LINE AVAILABLE FOR DATA?
NO
YES
MSTPRINT
CHANGE COLUMNS
RETURN

SUBORDINATE TERM EXTRACTION (RELTERM) PROCESSING LOGIC

The files are opened and the UPSI control data is scanned to determine which types of terms are to be extracted. If no UPSI card is present only related terms are to be extracted. A switch is set indicating the term types being extracted and a message is printed out. Then the first dictionary record is read. The contents position is tested to determine whether or not this record contains any of the term types to be extracted. If it does the record is scanned and each term which is a hit (i.e. has been requested) is written out along with its type and the main term with which it is associated. If there are no requested term types present the record is ignored. Then the next record is processed. As soon as an end of file is encountered on the dictionary tape processing terminates.

Input

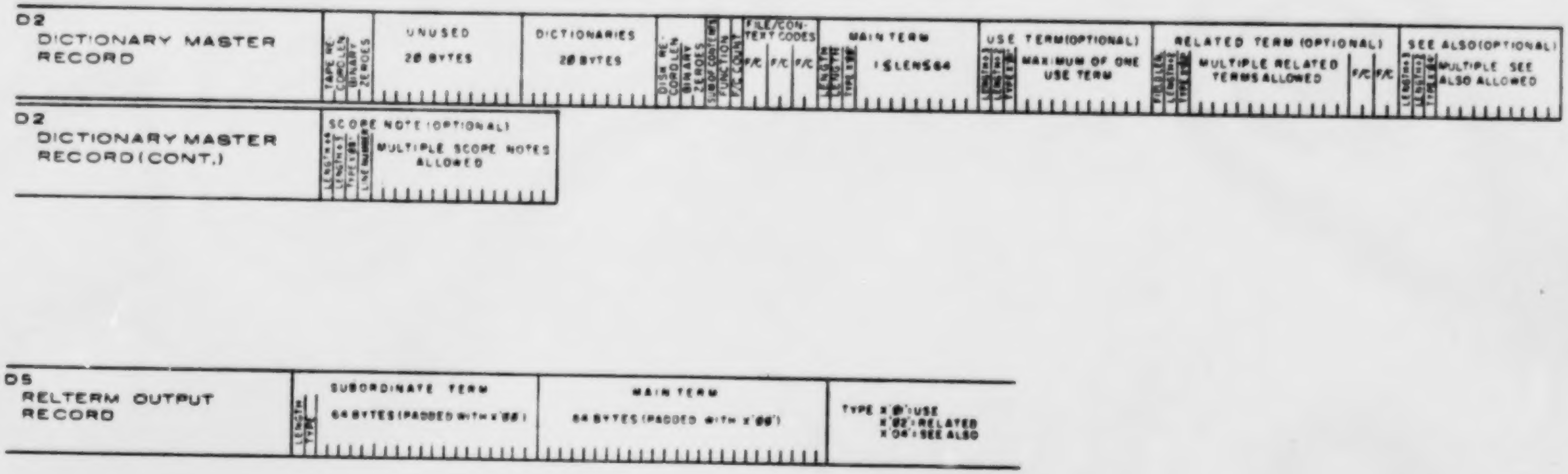
Input consists of the tape containing the current dictionary (D2 below) and a UPSI control card which specifies which types of subordinate terms are to be validated. The format of the card is as follows:

// UPSI 00000 $\underset{S}{\underline{1}}\ \underset{R}{\underline{1}}\ \underset{U}{\underline{1}}$, where a 1 in the S position indicates extraction of see also terms, a 1 in the R position causes extraction of related terms, and a 1 in the U position causes extraction of use terms.

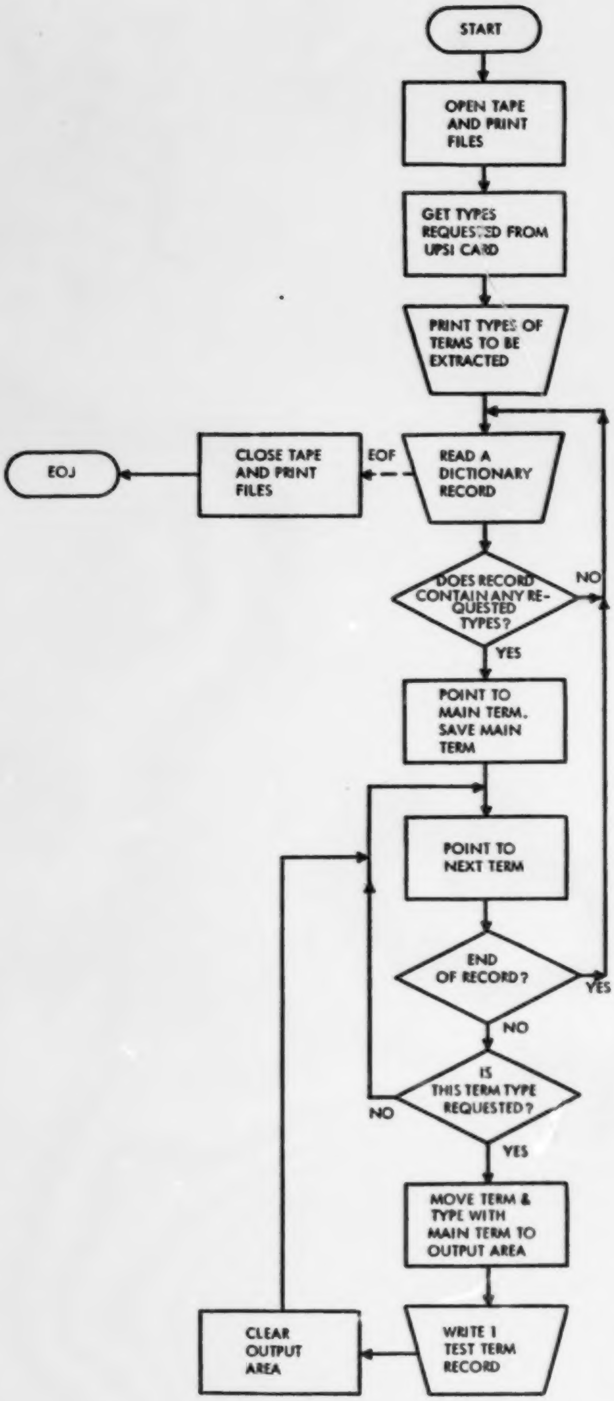
Output

The tape created by RELTERM is composed of fixed length blocked records in the format labeled D5 below. After sorting this tape will provide test input to RELTRM1.

SUBORDINATE TERM EXTRACTION (RELTERM) RECORD FORMATS


D2
DICTIONARY MASTER RECORD
TAPE RE-CORDLEN
BINARY ZEROES
UNUSED
20 BYTES
DICTIONARIES
20 BYTES
DISK RE-CORDLEN
BINARY ZEROES
FUNCTION
F/C COUNT
FILE/CON-TEXT CODES
F/C
F/C
F/C
MAIN TERM
LENGTH
1≤LEN≤64
USE TERM(OPTIONAL)
MAXIMUM OF ONE USE TERM
RELATED TERM (OPTIONAL)
MULTIPLE RELATED TERMS ALLOWED
F/C
F/C
SEE ALSO(OPTIONAL)
MULTIPLE SEE ALSO ALLOWED
D2
DICTIONARY MASTER RECORD(CONT.)
SCOPE NOTE (OPTIONAL)
MULTIPLE SCOPE NOTES ALLOWED
D5
RELTERM OUTPUT RECORD
LENGTH
TYPE
SUBORDINATE TERM
64 BYTES (PADDED WITH X'00')
MAIN TERM
64 BYTES (PADDED WITH X'00')
TYPE X'01':USE
X'02':RELATED
X'04':SEE ALSO

RELTERM
EXTRACT


START
OPEN TAPE AND PRINT FILES
GET TYPES REQUESTED FROM UPSI CARD
PRINT TYPES OF TERMS TO BE EXTRACTED
READ A DICTIONARY RECORD
EOF
CLOSE TAPE AND PRINT FILES
EOJ
DOES RECORD CONTAIN ANY RE-QUESTED TYPES?
NO
YES
POINT TO MAIN TERM, SAVE MAIN TERM
POINT TO NEXT TERM
END OF RECORD?
YES
NO
IS THIS TERM TYPE REQUESTED?
NO
YES
MOVE TERM & TYPE WITH MAIN TERM TO OUTPUT AREA
WRITE 1 TEST TERM RECORD
CLEAR OUTPUT AREA

SUBORDINATE TERM VALIDATION (RELTRM1) PROCESSING LOGIC

The files are opened and the first record is read from each tape. The main term is located on the dictionary tape and is compared to the term on the test tape. If the main term is alphabetically low the next record is read on the dictionary tape and that main term is compared. If the main term is alphabetically high an error switch is set. If the two terms are equal (i.e. the same) the error switch is cleared. The error switch is tested. If it has been set the entire test tape record (term, type, and main term) is written out on a work file. Then the next record is read from the test tape and the test term is compared to the test term in the preceding record. If it is the same term the error switch is tested and the record is processed as above. If the terms are different the term type is tested. If it is different from the type in the preceding record the dictionary tape is rewound and processing continues. If the types are the same the main term in the dictionary record is compared to the test term and processing continues as above.

As soon as an end of file is encountered on the test tape the error tape (work file written during the above phase) is read and printed. Each term is listed along with its type and the main term with which it occurs. As soon as an end of file is reached on the error tape processing terminates.

Input

Input consists of two magnetic tape files.

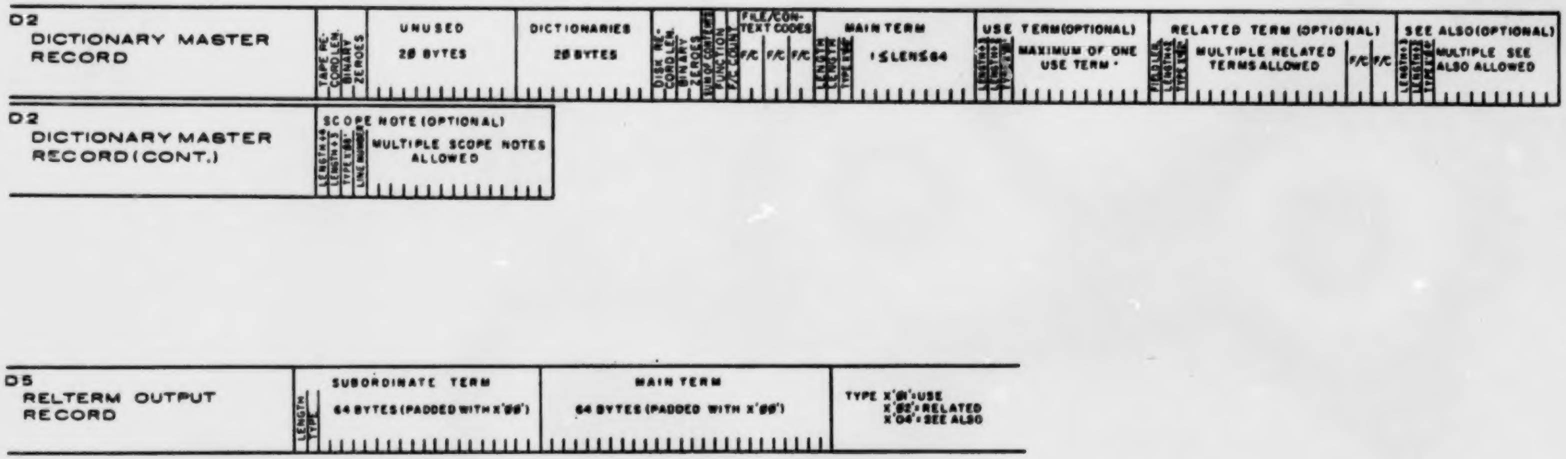
1. The tape containing the current dictionary (format D2 below) is read on SYS007.

2. The tape containing the sorted test terms is read from SYS005. The records are fixed length and blocked. The format is shown below labeled D5.

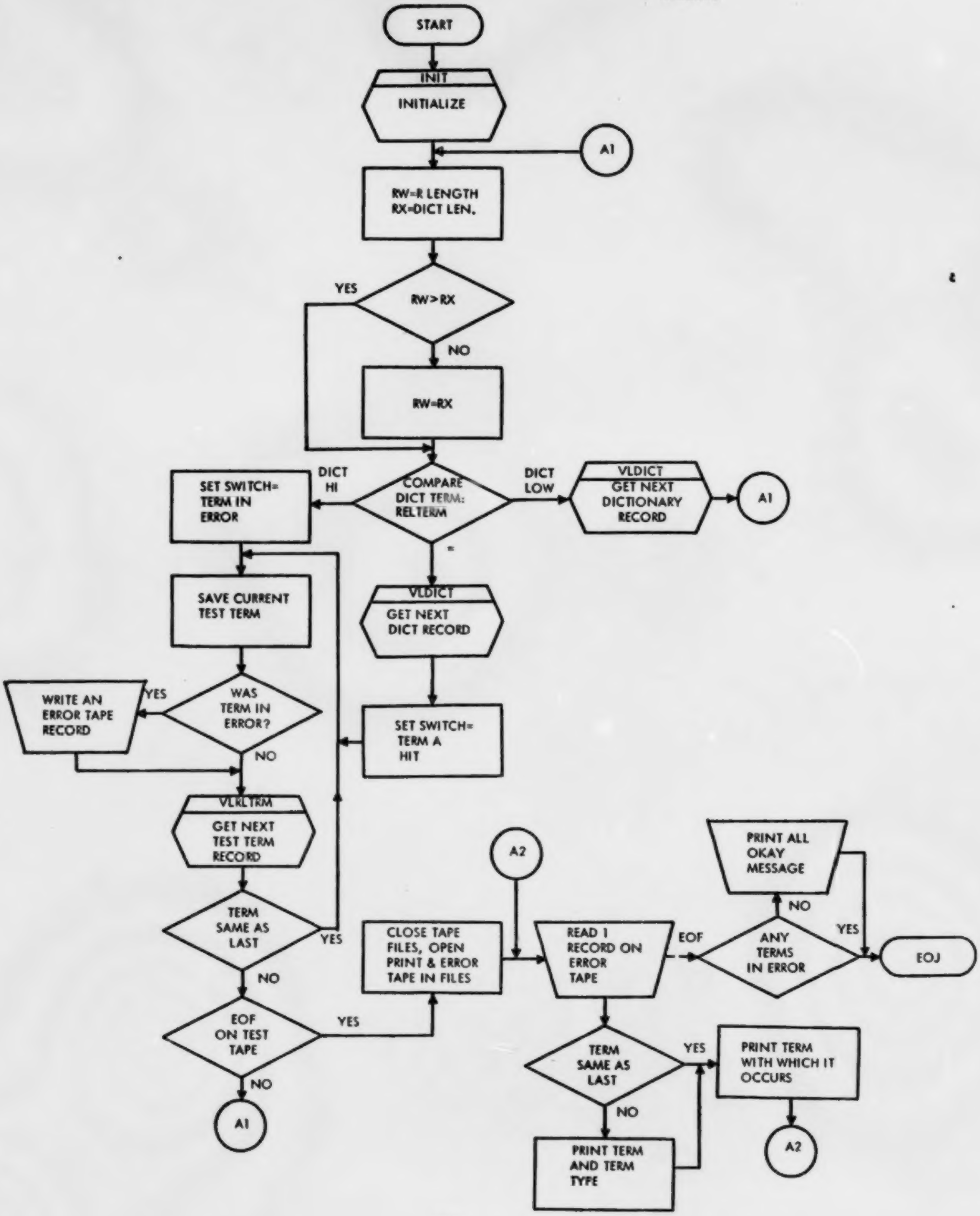
Output

A work tape in the D5 format below is written when missing terms are encountered. This tape is mounted on SYS004. The listing of erroneous terms is printed on SYSLST.

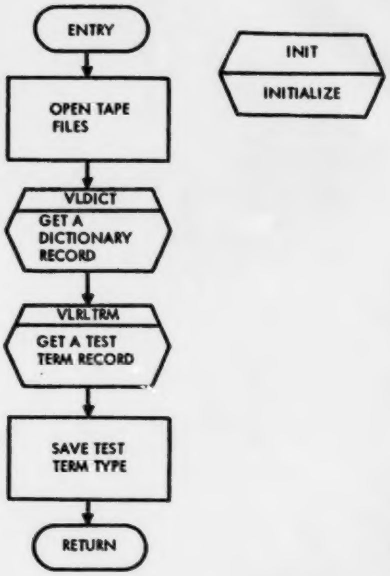
SUBORDINATE TERM VALIDATION (RELTRM1) RECORD FORMATS

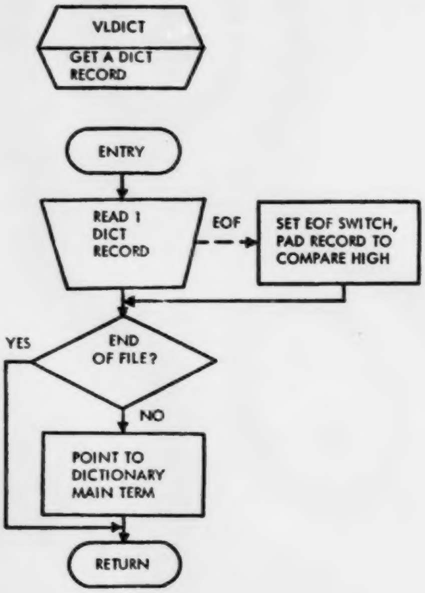

D2
DICTIONARY MASTER RECORD
TAPE RE-CORDLEN.
BINARY ZEROES
UNUSED
20 BYTES
DICTIONARIES
20 BYTES
DISK RE-CORDLEN.
BINARY ZEROES
FUNCTION
F/C COUNT
FILE/CON-TEXT CODES
F/C
F/C
F/C
MAIN TERM
LENGTH
LENGTH
TYPE
1≤LEN≤64
USE TERM(OPTIONAL)
MAXIMUM OF ONE USE TERM
RELATED TERM (OPTIONAL)
MULTIPLE RELATED TERMS ALLOWED
F/C
F/C
SEE ALSO(OPTIONAL)
MULTIPLE SEE ALSO ALLOWED
D2
DICTIONARY MASTER RECORD(CONT.)
SCOPE NOTE (OPTIONAL)
LENGTH+4
LENGTH+3
LINE NUMBER
MULTIPLE SCOPE NOTES ALLOWED
D5
RELTERM OUTPUT RECORD
LENGTH
TYPE
SUBORDINATE TERM
64 BYTES (PADDED WITH X'00')
MAIN TERM
64 BYTES (PADDED WITH X'00')
TYPE X'01':USE
X'02':RELATED
X'04':SEE ALSO

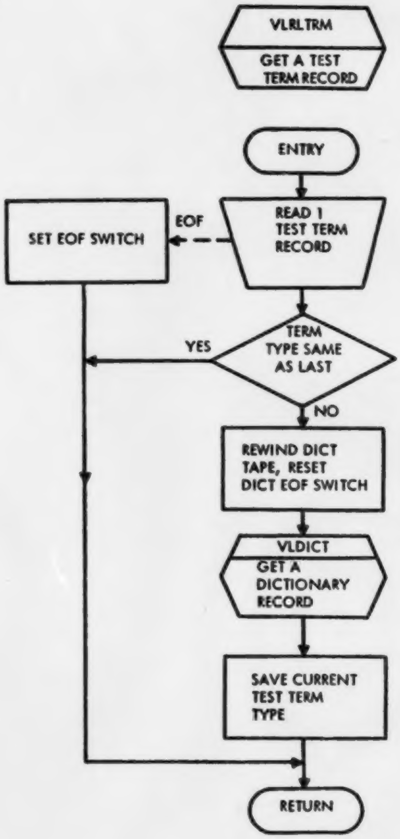
RELTRMI
VALIDATE


START
INIT
INITIALIZE
A1
RW=R LENGTH
RX=DICT LEN.
YES
RW > RX
NO
RW=RX
DICT
HI
SET SWITCH=
TERM IN
ERROR
COMPARE
DICT TERM:
RELTERM
DICT
LOW
VLDICT
GET NEXT
DICTIONARY
RECORD
A1
=
SAVE CURRENT
TEST TERM
VLDICT
GET NEXT
DICT RECORD
WRITE AN
ERROR TAPE
RECORD
YES
WAS
TERM IN
ERROR?
NO
SET SWITCH=
TERM A
HIT
VLRLTRM
GET NEXT
TEST TERM
RECORD
A2
PRINT ALL
OKAY
MESSAGE
TERM
SAME AS
LAST
YES
NO
CLOSE TAPE
FILES, OPEN
PRINT & ERROR
TAPE IN FILES
READ 1
RECORD ON
ERROR
TAPE
EOF
ANY
TERMS
IN ERROR
NO
YES
EOJ
EOF
ON TEST
TAPE
YES
NO
A1
TERM
SAME AS
LAST
YES
NO
PRINT TERM
WITH WHICH IT
OCCURS
A2
PRINT TERM
AND TERM
TYPE

RELTRM1
VALIDATE


INIT
INITIALIZE
ENTRY
OPEN TAPE FILES
VLDICT
GET A DICTIONARY RECORD
VLRLTRM
GET A TEST TERM RECORD
SAVE TEST TERM TYPE
RETURN


VLDICT
GET A DICT RECORD
ENTRY
READ 1 DICT RECORD
EOF
SET EOF SWITCH, PAD RECORD TO COMPARE HIGH
END OF FILE?
YES
NO
POINT TO DICTIONARY MAIN TERM
RETURN


VLRLTRM
GET A TEST TERM RECORD
ENTRY
READ 1 TEST TERM RECORD
EOF
SET EOF SWITCH
TERM TYPE SAME AS LAST
YES
NO
REWIND DICT TAPE, RESET DICT EOF SWITCH
VLDICT
GET A DICTIONARY RECORD
SAVE CURRENT TEST TERM TYPE
RETURN

MASTER FILE MAINTENANCE EDIT PROGRAM (MFEDIT) PROCESSING LOGIC

1. Read input record and check if it is HDR FILE record. If not, print error and go back to 1. Otherwise, save file number from HDR FILE record. Read next input record.

2. Check if it is HDR ADD, HDR REV or HDR DEL. If not, print error. Read next input record and go back to 2. Otherwise, get internal or external number from input record and check if it is valid. Print error, if not. Otherwise create a HDR record and hold it in work area.

2a. Read next input record.

3. If the Type is HDR, go to 4. If the Type is Title, get title information from the record, and create and hold a REV TITLE record. Go to 2a. If the type is ADD, scan for next colon and go to 5. If the type is DELETE, scan for semi-colon and go to 6. Otherwise print error and go to 2a.

4. Test if any errors except warning errors were printed for changes to the previous document. If yes, go to 2. Otherwise check if a document was added without at least one descriptor. If yes, go to 2. Otherwise,output all records pertaining to the previous document from work area. Go to 2.

5. Test if Type is T. If yes, read input records and build a free text segment record, until a record with a non blank operation is read. Then go to 2. Otherwise, test if type is D. If no, print error and go to 2a. Otherwise, check if anything in work area for related terms. If yes, create and hold ADD descriptor records for each one. Otherwise, get descriptor to be added. Look up this descriptor in the dictionary. Print error if not found. Otherwise, create and hold an ADD DESCRIPTOR record. Hold related terms for this descriptor, if any, in a work area for related terms.

5a. Scan for next colon.

5b. If Type is S, go to 5i.

5c. If Type +1 is V, go to 5h.

5d. If Type is LK, go to 5g.

5e. If Type is not C, go to 5. Otherwise, test if categories have been added to this descriptor. Print error if yes.

5f. Get category and hold it in a work area. Scan for next colon. Test if Type is another C. If yes, go to 5f. Otherwise, sort the categories collected. Create and hold an ADD CATEGORIES record. Go to 5b.

5g. Test if links have been added to this descriptor. Print error if yes. Collect links from the input record. Build a two byte mask in which each 1 bit represents the link value. Create and hold ADD LINKS record. Go to 5a.

5h. Get value. Convert to internal format. If it is a single value, move it into the work area. If it is a lower value, get next value, and test if it is an upper value. Print error if not. Otherwise, convert to internal format and test if it is higher than the lower value. Print error if not. Otherwise, move the value pair into the work area. Scan for next colon and test if it is another V. If yes, go to 5h. Otherwise, create and hold an ADD values record. Go to 5b.

5i. Get subdescriptor to be added. Look it up in the dictionary. Print error if not found. Otherwise, create and hold an ADD SUBDESCRIPTOR record. Go to 5a.

6. Test if TYPE is a D. If not, go to 6a. Otherwise, get descriptor from input record. Look it up in the dictionary. Print error if not found. Otherwise create and hold a DELETE DESCRIPTOR record. Scan for next colon and go to 3.

6a. Test if TYPE is S. If not, go to 6d.

6b. Otherwise, get descriptor from input record. Look it up in the dictionary. Print error if not found. Otherwise, create and hold a LOCATOR DESCRIPTOR record. This record will be used to locate the descriptor from input record. Look it up in the dictionary. Print error if not found and go to 6c. Otherwise, create and hold a DELETE SUBDESCRIPTOR record.

6c. Scan for next colon. Test if TYPE is a D. If yes, go to 6b. Otherwise, go to 2a.

6d. Test if TYPE is V. If not, go to 6f. Otherwise, get descriptor from input record, validate it and create and hold LOCATOR DESCRIPTOR record. Scan for next colon. Test if TYPE is S. If not, go to 6e. If yes, get subdescriptor from input record, validate it, and create and hold a LOCATOR SUBDESCRIPTOR record. The LOCATOR DESCRIPTOR and/or LOCATOR SUBDESCRIPTOR records will be used to locate the descriptor and/or subdescriptor that the values to be deleted are associated with. Scan for next colon.

6e. Test if TYPE is V. If not, print error and go to 2a. Otherwise, collect and check values to be deleted until TYPE is not V. Then create a DELETE VALUES record. Go to 3.

6f. Test if TYPE is LK. If not, go to 6h.

6g. Otherwise, get descriptor from input, validate against the dictionary, create and hold a LOCATOR DESCRIPTOR record. Get links, build a two byte mask and create and hold a DELETE LINKS record. Scan for next colon. Test if TYPE is D. If yes, go to 6g. Otherwise, go to 2a.

6h. Test if Type is C. If no, go to 6i. Otherwise, get descriptor from input record, validate it against the dictionary, create and hold a LOCATOR DESCRIPTOR record. Collect categories to be deleted until Type is not C. Then create a DELETE CATEGORIES record. Go to 3.

6i. Test if Type is T. If no, print error, and go to 2a. Otherwise, collect text segments and create and hold a DELETE TEXT record. Go to 3.

The program keeps reading input records and processing them in the above manner until there are no more cards or it reads an end of file on the input tape. The program is then terminated.

CNVRT

This routine, accessed by VSCAN, converts specified values to binary mantissas and exponents.

Once the nine character number area and the six character value area have been cleared to zeroes, a check is made to see if a time-value, date value or other value is to be converted. The proper routine for checking the type of value present is entered. If no error is logged in this routine, control is returned to CNVRT and checking continues. If a number other than zero was found, it is packed (de-zoned) and converted to binary. This is then stored in the last four positions of the value area. If there is an exponent associated with the number it, too, is checked. If the exponent consisted of between one and three digits and was not greater than 255, it is converted to binary. If a negative exponent was indicated, the value is complemented. The normalization factor is then added to the exponent. This number is checked to see that it falls between 0-255 inclusive. The exponent is moved to the record position of the value area, and the sign of the number is 'OR'ed with it. The code for the type of value processed is moved to the type byte (first byte) of the value area.

EOFRTN

When the end of file record is read, EOFRTN is entered. Any error messages logged since the last HDR card are printed. If there were no serious errors the last output records are written on tape. Last, all files are closed.

ERRDATA

This is used only for values. The portion of the input record from the beginning of the value information up to the place where an error was discovered is moved to the alternate print buffer to be double printed along with the remainder of the input record.

FILECK

This routine is used to validate file and context entries. It takes the FIL mnemonic from the HDR FILE card and CXT entries from any subsequent cards, adds trailing blanks if three characters were not specified, and checks this three character code against the proper table. There is one file table and as many context tables as there are defined files in the file table. The file table is in core at all times. The context table in core is the one which corresponds to the file mnemonic on the HDR FILE: card. Only those CXT entries which are related to this file are checked. All descriptors processed by the edit must have at least one such context.

A binary search is used to match the input entry with a table entry. If a match is made the binary number which corresponds to the entry is saved. If no match is found, an error bit is set.

COMMA

This routine compresses descriptive information. It can be entered for two reasons. First, if there is a DELETE card, there must be a one character code followed by a semi-colon to indicate the type of delete information which will follow. Second, if there is a colon, there must be further information followed either by a semi-colon or a non-continuation card.

The input information is checked one character at a time for either blank, semi-colon or non-blank. If the character is a blank, the "blank" switch is set. If it is non-blank, and not a semi-colon, the "blank" switch is tested to see if it is set. If it is, one blank is moved to the work area, and the blank switch turned off. The character found is then moved to the work area. If a semi-colon was encountered, a switch is set. The number of characters placed in the work area is calculated and stored, and a return is made to the calling routine.

If a full input record is checked and no semi-colon is found, the next record is read. If it is a continuation card, checking continues. Otherwise, the number of characters placed in the work area is calculated and stored, and a return is made to the calling routine.

There may be only 64 characters moved to the work area. If a 65th character is found, an error is logged. Checking continues, but no more characters are moved.

SCAN

This is used for descriptors and subdescriptors only when delete records are being built. It checks to see that the expected type and specified type are the same.

SORT

This sorts the binary context numbers which have been validated and stored. While arranging the numbers in ascending sequence, the sort checks for duplicate entries. If any are found, a warning message is logged, and the duplicate entry is eliminated from the list.

TOTAPE

This writes on tape the records which have been built in core. The records are variable and are put out in blocks of 3625.

DVSCAN

This checks to see that date values are entered in the form month-day-year. Month may be one or two digits and may not be greater than 12. Day may be one or two digits and may not be greater than 31. Year may be two or four digits and may not be greater than 2100. If only two digits are given for year, a "19" is prefixed to them.

NUMSCAN

This routine checks for the sign, decimal point and exponent'E' of numbers being validated. It also checks to see that the number has no imbedded blanks and that it consists only of digits.

The general form of the number should be: ± NNN.NNN E NNN;. There does not have to be a sign, decimal point or exponent. The mantissa may contain up to nine digits with a decimal point anywhere. The exponent can have only three digits and may have no decimal point. If more than one sign, or an imbedded sign, if found; if more than one decimal point is found, if more than one exponent 'E' is found, if more than nine digits are found; or if a non-numberic, non-valid special character is found, an error is logged.

PUT

This routine calculates the length of the last record built in core and stores it in front of the record. It then saves the address of the length byte for the next record to be built, and positions the output buffer pointer at the next type byte. If the last record built would have extended beyond the end of the 3625 byte buffer, the record is moved to the beginning of a new buffer. The block length of the last buffer is calculated and stored.

READ

The end of file switch is checked. If it is on, there is a branch to EOFRTN. Otherwise the input record which has just been scanned is sequence checked. If it is not high, an error is logged. The sequence number on this card is saved. The card is moved to the print buffer where colons and semi-colons are changed to slashes and commas. The modified record is then printed with a space supress. Next, the alternate buffer, the one which repeats any values which were in error, is moved to the print buffer. Again, colons and semi-colons are changed to slashes and commas. This is then printed over the input record so that any values containing errors will be double-printed. The alternate buffer is now blanked.

The next record is read. If it is an end of file, a switch indicating this is set and the read routine is exited. If it is a continuation card, the routine is exited. If it is a new type card, the old card type is saved and the new type is moved to a hold area. The routine is exited.

SLASH

Characters from the input buffer are moved one at a time, ignoring blanks, into a field called TYPE, until a colon is encountered. An error is printed if more than six characters are moved into TYPE. If the end of buffer is reached before a colon is encountered, the next input record is read and checked for blank operation. If the operation is blank, this record is considered to be a continuation of the previous one and scanning continues. The subroutine returns to the calling program either upon encountering a colon or reading a record with non-blank operation field.

VALIDNO

This subroutine is used to validate segment numbers, category numbers, rank numbers, file numbers. The subroutine collects one to three digits from the input record. Then it converts these digits to binary and tests if the result is greater than zero and less than or equal to 255. Then it returns to the calling program.

VSCAN

This subroutine determines which type of value is present, i.e., single value, lower value, upper value, time value or date value. It also checks whether lower value and upper values, if present, occur in consecutive pairs and lower value is actually less than upper value. The routine then converts these values to internal format and returns to the calling program.

VALUECK

This subroutine compares two values (say V1 and V2) in the internal format and sets the condition code to indicate the result of the compare (i.e., whether VI is higher than, equal to or lower than V2). The subroutine returns to the calling program.

TVSCAN

This subroutine is used to move a time value (specified as HH.MM.SS. i.e. hours, minutes and seconds) from the input buffer to a field called TIME. The decimal points are not moved into TIME. Hours, minutes and seconds are checked to see if they exceed 24, 60, 60 respectively. Errors are printed if they do and the subroutine returns to the calling program. A normal return is made if no errors are encountered.

VALIDITY

This routine validates descriptors and subdescriptors. The term name is checked to see that it contains only alpha-numeric characters, blanks, and allowable special characters. If the first character of the term is a $, the term is not looked up in the dictionary. Otherwise, the lookup is performed. If the term was not found in the dictionary, an error is logged and the routine is exited. If it was found and if it was a descriptor, the dictionary record is checked to see if this term has a preferred term (USE) associated with it. If it does, this term is looked and used instead of the entered term. If the USE term also has a USE term, it is in error, and the routine is exited.

If the term was a valid descriptor, its related terms their FIL-CXT codes are moved from the dictionary record to a core storage area.

The first five characters of both descriptors and subdescriptors are checked for TIME or DATE. If the term begins with one of these, a switch is set to indicate that a time-value or date-value is acceptable.

MESSPRT

Whenever a HDR record, first SCOPENNN record of a series, or an EOF record is encountered, a check of the error message bytes is made. Each bit in the sixteen bytes corresponds to an error note. Anytime a bit is found to be on (i.e., a '1') the corresponding error message is printed. If any bit of the first fifteen is on, no records will be written for the term just processed. A switch (NOREC) is set to indicate this. Bits of the last error message bytes correspond to warning error messages. These do not prevent output from being written. Before the routine is exited, the error message bytes are all set to X'00'.

NOTENUM

This routine shares logic with MESSPRT. Its function is to indicate to the user the data set in which an error occurred.

Whenever a non-continuation card is encountered (other than the 2-n scope notes in a series) a check of the error number bytes is made. If a bit is found to be on, the number of the corresponding message is placed in the print buffer. These numbers are printed 25 at a time.

When the routine is entered, the error number bytes are "OR"ed with the error message bytes. When the routine is exited the error number bytes are set to all X'00'.

Once all the message error bytes have been checked, NOREC is tested. If it is not set, tape records are now formed and written. Each of the 79 character partial records formed in FORMREC is moved to the tape output work area immediately after the record byte (X'01'), length byte for the main term, and the main term name padded with blanks to 64 positions. The completed 145 character output record is then written. A X'FF' in the partial record hold area indicates that all records have been formed.

DMK01 (Dictionary Disk Look-Up Routine)

During initialization the communication vector is read from cylinder 1, track 0 of the first dictionary disk file into core. If the condensed dictionary is not mounted, the note "MOUNT FDA DICT DISK" is typed on the console typewriter. The job will continue after the correct disk is mounted. When searching for a term, first the master index is scanned to find which cylinder index to scan. If this cylinder index is not already in core, it is read into core. The selected cylinder index table is scanned to find which cylinder to scan. If the track index for this cylinder is not in core, it is read to core. The selected track index is scanned to find which track to scan. If the selected track is not in core it is read to core. As this track is scanned, if there is a track index or a record which is continued from the last track, this record is bypassed. If the term being looked up is not on this track, the first location of the output area is flagged as not found (X'FF'). Otherwise, the entire condensed dictionary record is moved to the output area. If this record is continued on the next track, the next track is read and the rest of the condensed dictionary record is put in the output area.

This same disk look-up routine is used by various other programs in this system.

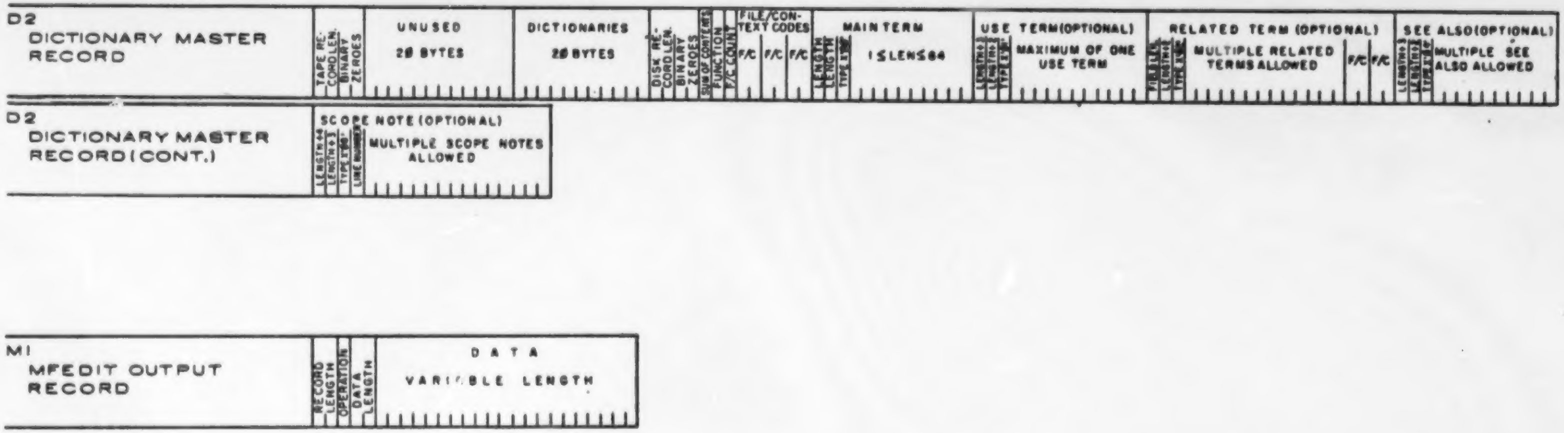
Input

Input is from either cards or a magnetic tape containing card images. For a detailed description of the various cards accepted by MFEDIT, please refer to the file maintenance language section of the User's Manual. The dictionary disk record (D2 below) is used in validating input. The disk record does not include the first 44 bytes shown in the layout.

Output

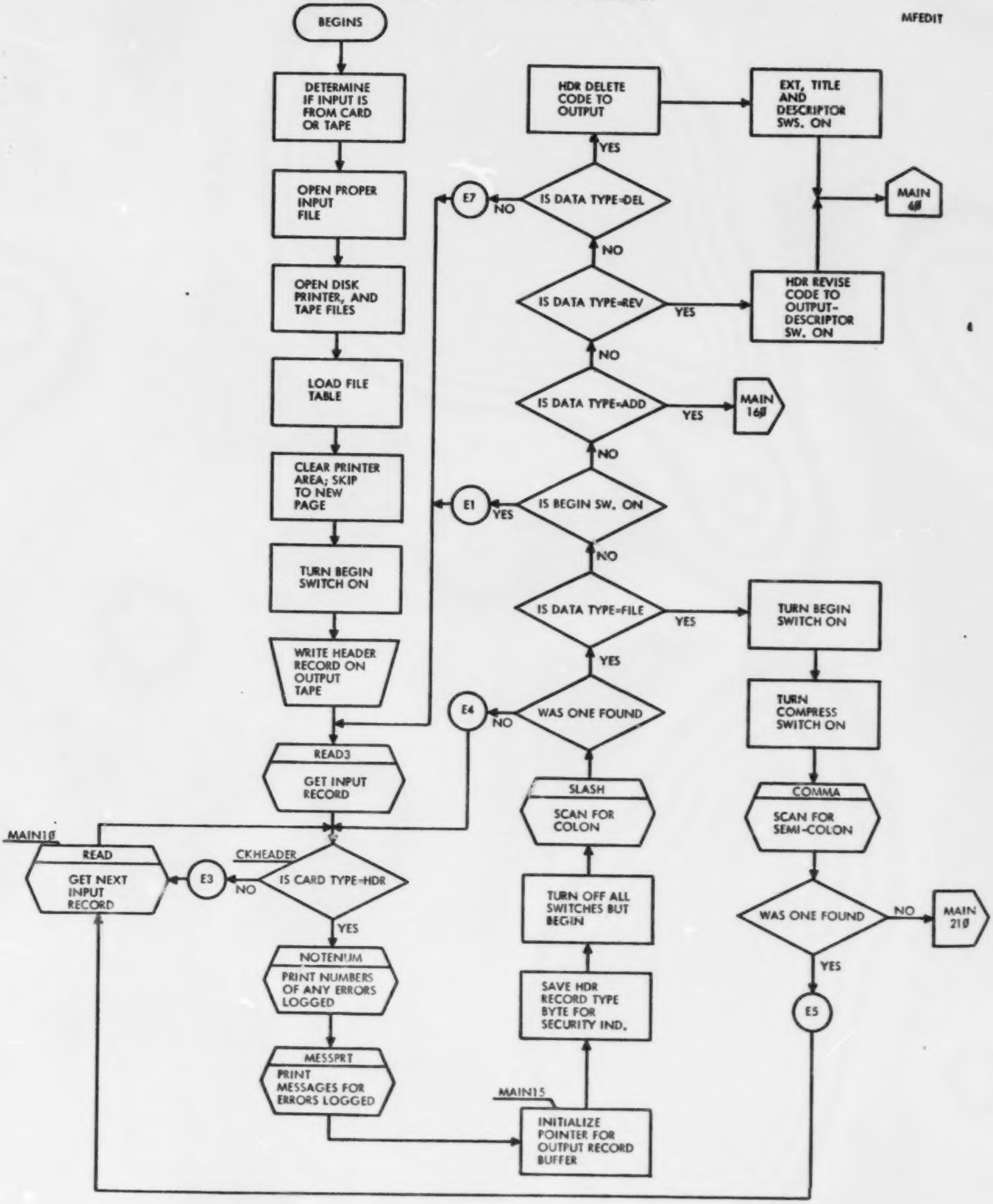
MFEDIT creates a tape containing data which will provide input to MFMAINT. Its format is shown below labelled M1.

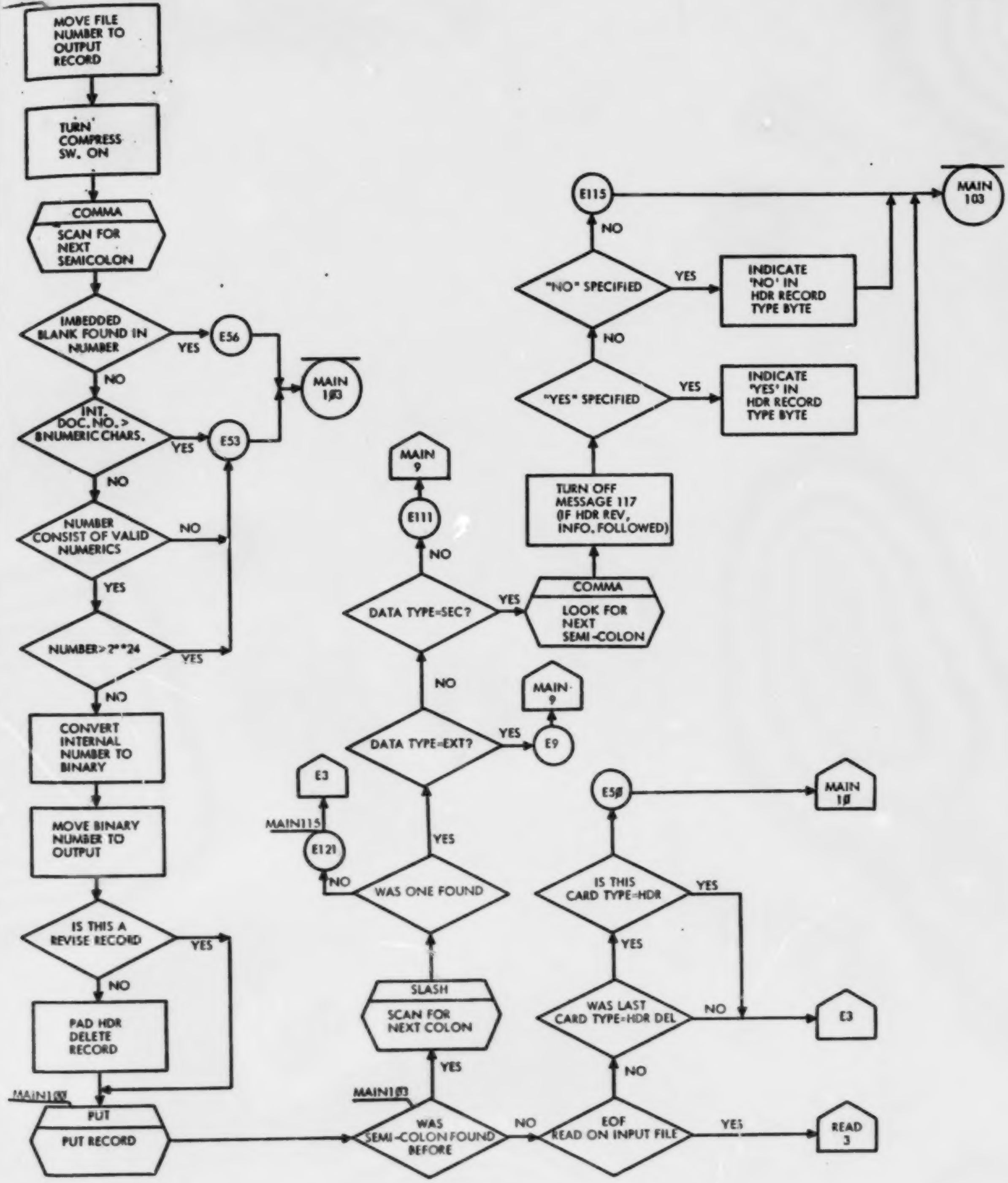
MASTER FILE MAINTENANCE EDIT PROGRAM (MFEDIT) RECORD FORMATS

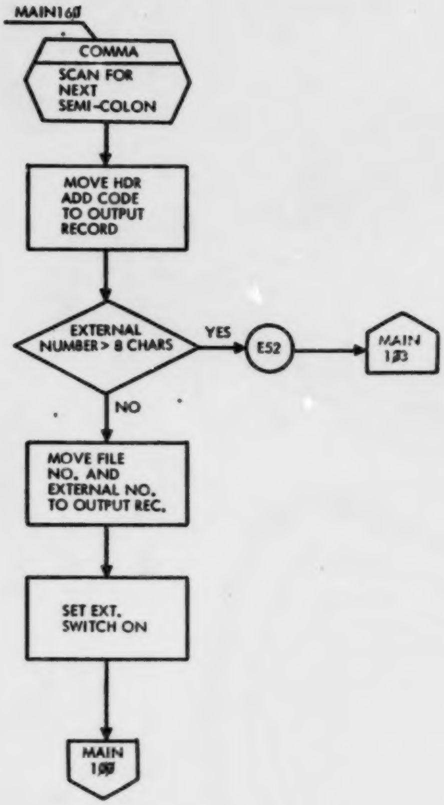

D2
DICTIONARY MASTER RECORD
TAPE RE-CORDLEN.
BINARY ZEROES
UNUSED
2Ø BYTES
DICTIONARIES
2Ø BYTES
DISK RE-CORDLEN.
BINARY ZEROES
SUM OF CONTENTS
FUNCTION
F/C COUNT
FILE/CON-TEXT CODES
F/C
F/C
F/C
MAIN TERM
LENGTH
LENGTH
1≤LEN≤64
USE TERM(OPTIONAL)
MAXIMUM OF ONE USE TERM
RELATED TERM (OPTIONAL)
MULTIPLE RELATED TERMS ALLOWED
F/C
F/C
SEE ALSO(OPTIONAL)
MULTIPLE SEE ALSO ALLOWED
D2
DICTIONARY MASTER RECORD(CONT.)
SCOPE NOTE (OPTIONAL)
LENGTH+4
LENGTH+3
LINE NUMBER
MULTIPLE SCOPE NOTES ALLOWED
M1
MFEDIT OUTPUT RECORD
RECORD LENGTH
OPERATION
DATA LENGTH
DATA
VARIABLE LENGTH

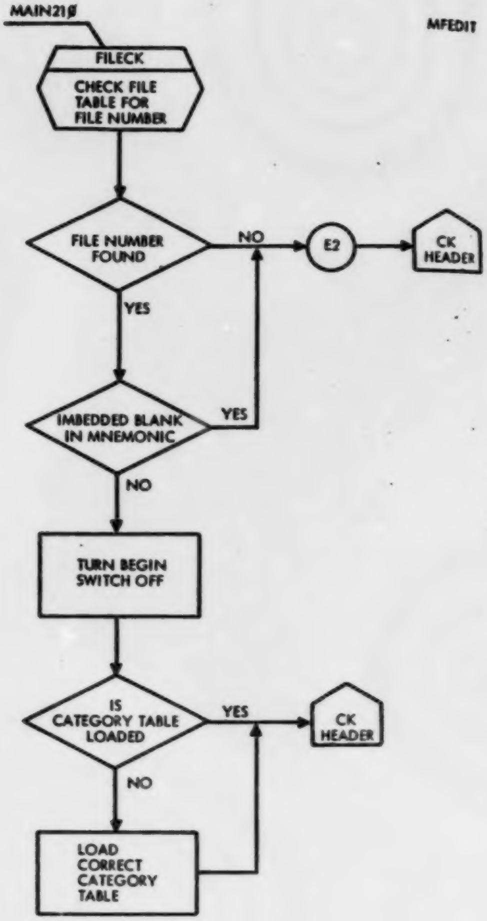
FMEDIT-EDIT OF FILE MAINTENANCE INPUT

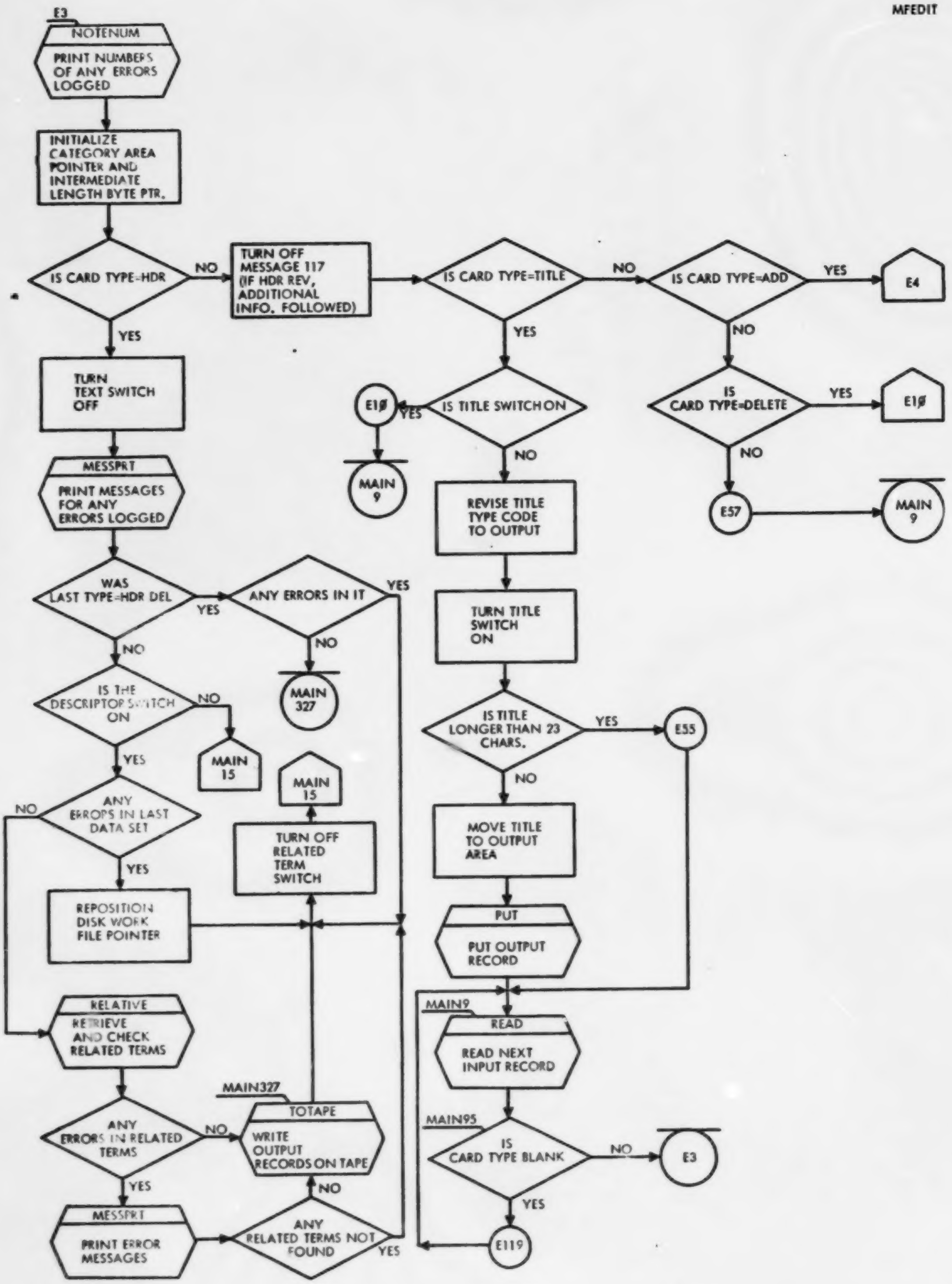
MFEDIT

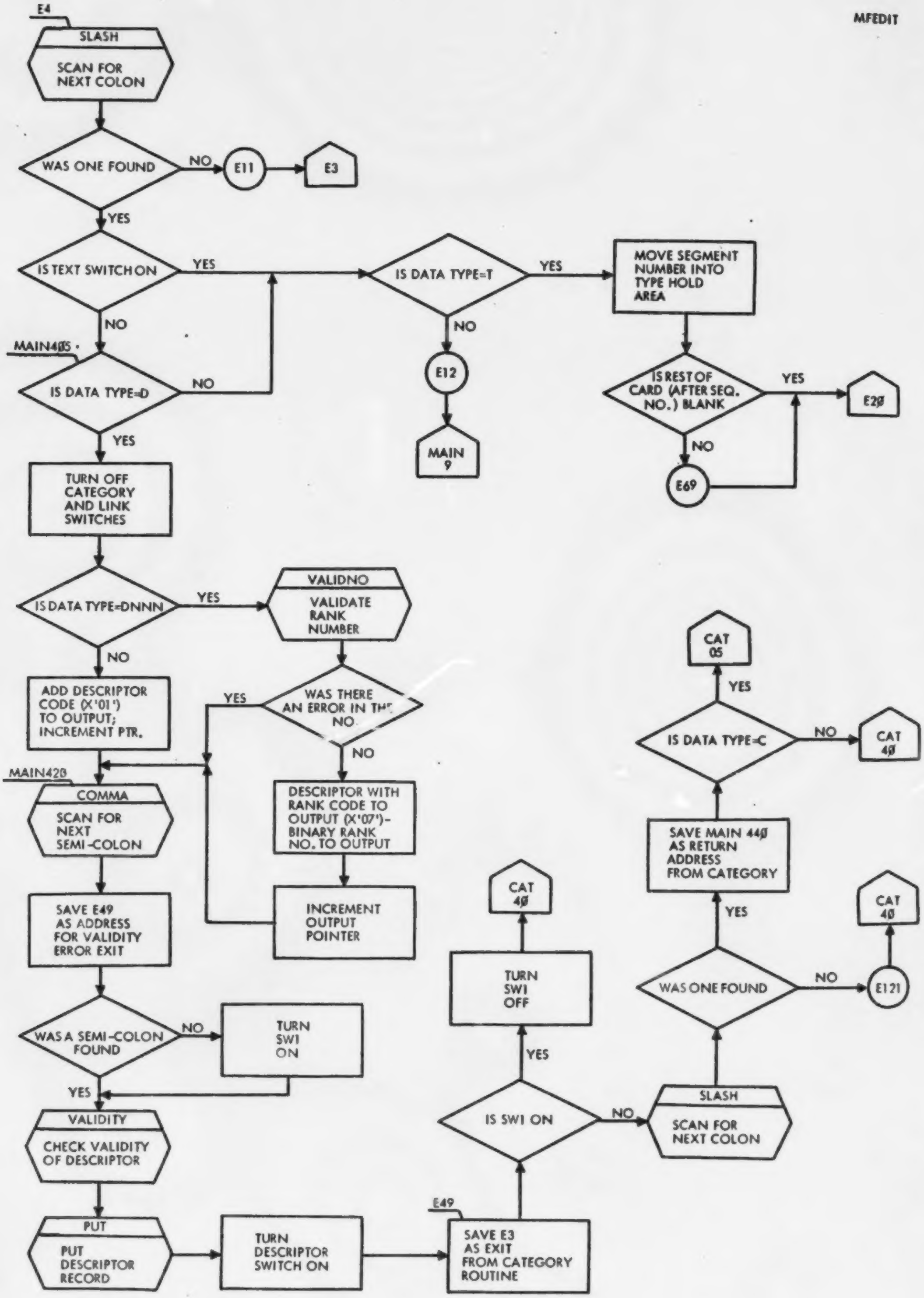

BEGINS
DETERMINE IF INPUT IS FROM CARD OR TAPE
OPEN PROPER INPUT FILE
OPEN DISK PRINTER, AND TAPE FILES
LOAD FILE TABLE
CLEAR PRINTER AREA; SKIP TO NEW PAGE
TURN BEGIN SWITCH ON
WRITE HEADER RECORD ON OUTPUT TAPE
READ3
GET INPUT RECORD
MAIN10
READ
GET NEXT INPUT RECORD
E3
NO
CKHEADER
IS CARD TYPE=HDR
YES
NOTENUM
PRINT NUMBERS OF ANY ERRORS LOGGED
MESSPRT
PRINT MESSAGES FOR ERRORS LOGGED
MAIN15
INITIALIZE POINTER FOR OUTPUT RECORD BUFFER
SAVE HDR RECORD TYPE BYTE FOR SECURITY IND.
TURN OFF ALL SWITCHES BUT BEGIN
SLASH
SCAN FOR COLON
WAS ONE FOUND
NO
E4
YES
IS DATA TYPE=FILE
YES
NO
IS BEGIN SW. ON
YES
E1
NO
IS DATA TYPE=ADD
YES
MAIN 160
NO
IS DATA TYPE=REV
YES
NO
IS DATA TYPE=DEL
NO
E7
YES
HDR DELETE CODE TO OUTPUT
EXT, TITLE AND DESCRIPTOR SWS. ON
MAIN 60
HDR REVISE CODE TO OUTPUT-DESCRIPTOR SW. ON
TURN BEGIN SWITCH ON
TURN COMPRESS SWITCH ON
COMMA
SCAN FOR SEMI-COLON
WAS ONE FOUND
NO
MAIN 210
YES
E5

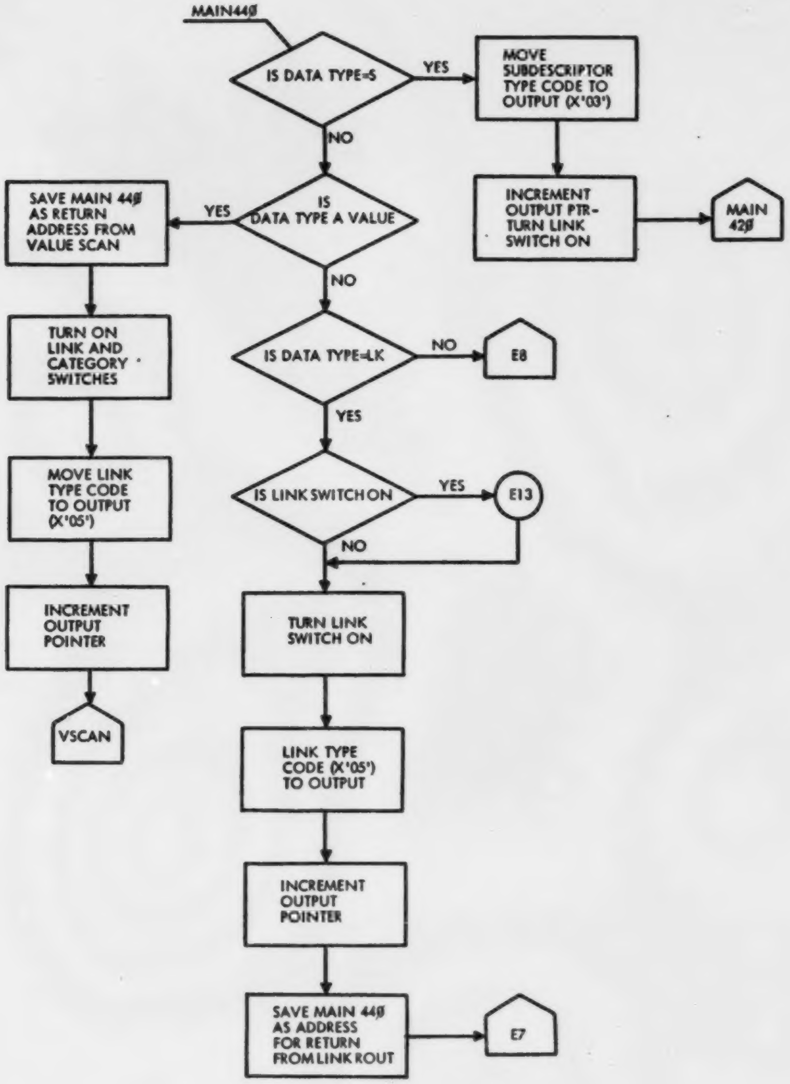

MOVE FILE NUMBER TO OUTPUT RECORD
TURN COMPRESS SW. ON
COMMA
SCAN FOR NEXT SEMICOLON
IMBEDDED BLANK FOUND IN NUMBER
YES
E56
NO
MAIN 103
INT. DOC. NO. > 8NUMERIC CHARS.
YES
E53
NO
NUMBER CONSIST OF VALID NUMERICS
NO
YES
NUMBER>2**24
YES
NO
CONVERT INTERNAL NUMBER TO BINARY
MOVE BINARY NUMBER TO OUTPUT
IS THIS A REVISE RECORD
YES
NO
PAD HDR DELETE RECORD
MAIN100
PUT
PUT RECORD
MAIN103
WAS SEMI-COLON FOUND BEFORE
NO
YES
SLASH
SCAN FOR NEXT COLON
WAS ONE FOUND
NO
E121
MAIN115
E3
YES
DATA TYPE=EXT?
YES
E9
MAIN 9
NO
DATA TYPE=SEC?
NO
E111
MAIN 9
YES
COMMA
LOOK FOR NEXT SEMI-COLON
TURN OFF MESSAGE 117 (IF HDR REV. INFO. FOLLOWED)
"YES" SPECIFIED
YES
INDICATE 'YES' IN HDR RECORD TYPE BYTE
NO
"NO" SPECIFIED
YES
INDICATE 'NO' IN HDR RECORD TYPE BYTE
NO
E115
MAIN 103
EOF READ ON INPUT FILE
YES
READ 3
NO
WAS LAST CARD TYPE=HDR DEL
NO
E3
YES
IS THIS CARD TYPE=HDR
YES
E50
MAIN 10


MAIN160
COMMA
SCAN FOR NEXT SEMI-COLON
MOVE HDR ADD CODE TO OUTPUT RECORD
EXTERNAL NUMBER > 8 CHARS
YES
E52
MAIN 103
NO
MOVE FILE NO. AND EXTERNAL NO. TO OUTPUT REC.
SET EXT. SWITCH ON
MAIN 100

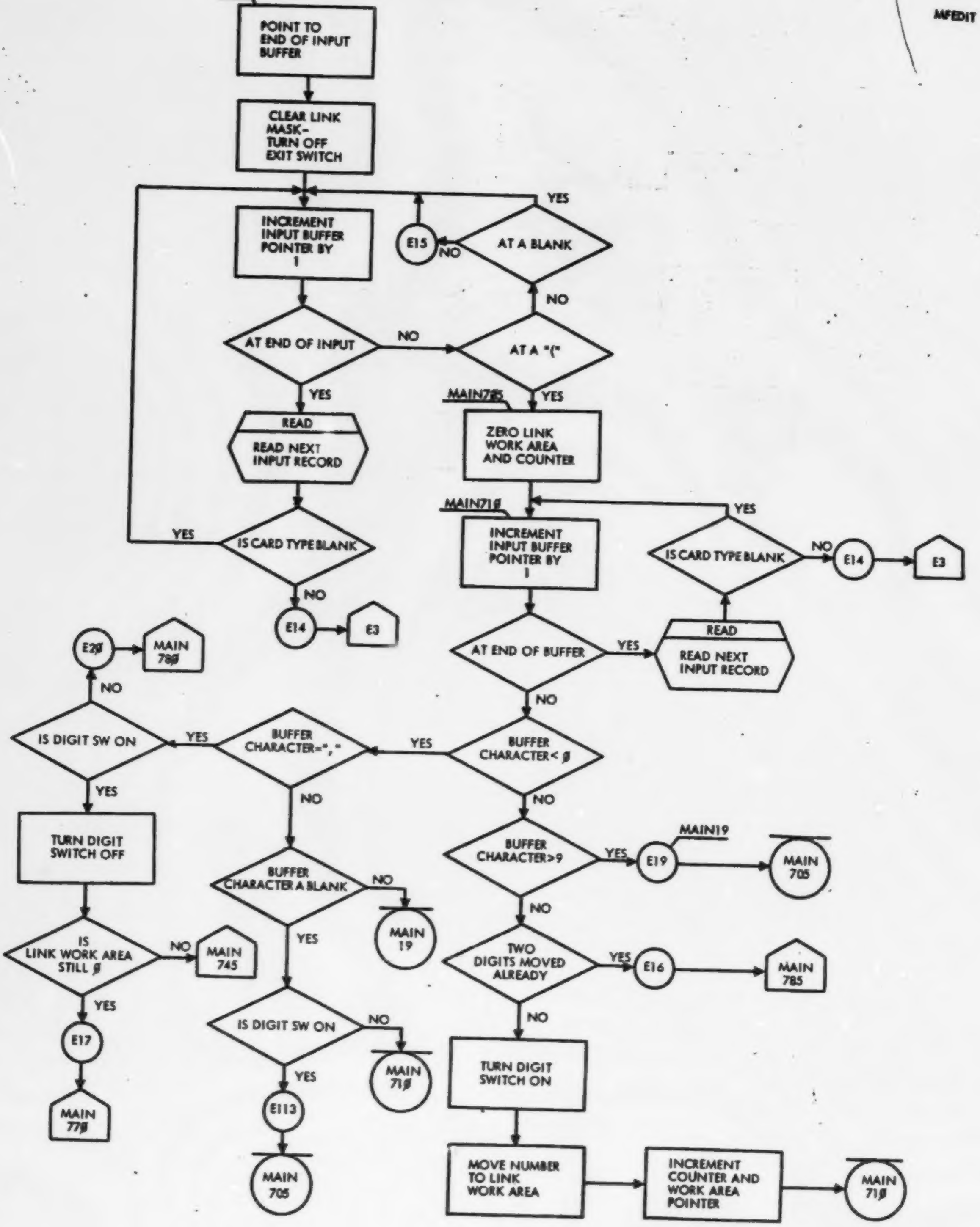

MAIN210
FILECK
CHECK FILE TABLE FOR FILE NUMBER
FILE NUMBER FOUND
NO
E2
CK HEADER
YES
IMBEDDED BLANK IN MNEMONIC
YES
NO
TURN BEGIN SWITCH OFF
IS CATEGORY TABLE LOADED
YES
CK HEADER
NO
LOAD CORRECT CATEGORY TABLE


E3
NOTENUM
PRINT NUMBERS OF ANY ERRORS LOGGED
INITIALIZE CATEGORY AREA POINTER AND INTERMEDIATE LENGTH BYTE PTR.
IS CARD TYPE=HDR
NO
TURN OFF MESSAGE 117 (IF HDR REV, ADDITIONAL INFO. FOLLOWED)
IS CARD TYPE=TITLE
NO
IS CARD TYPE=ADD
YES
E4
YES
TURN TEXT SWITCH OFF
YES
NO
E10
YES
IS TITLE SWITCH ON
IS CARD TYPE=DELETE
YES
E10
NO
NO
MAIN 9
REVISE TITLE TYPE CODE TO OUTPUT
E57
MAIN 9
MESSPRT
PRINT MESSAGES FOR ANY ERRORS LOGGED
WAS LAST TYPE=HDR DEL
YES
ANY ERRORS IN IT
YES
TURN TITLE SWITCH ON
NO
NO
MAIN 327
IS THE DESCRIPTOR SWITCH ON
NO
MAIN 15
IS TITLE LONGER THAN 23 CHARS.
YES
E55
YES
MAIN 15
NO
NO
ANY ERRORS IN LAST DATA SET
TURN OFF RELATED TERM SWITCH
MOVE TITLE TO OUTPUT AREA
YES
REPOSITION DISK WORK FILE POINTER
PUT
PUT OUTPUT RECORD
RELATIVE
RETRIEVE AND CHECK RELATED TERMS
MAIN9
READ
READ NEXT INPUT RECORD
MAIN327
TOTAPE
MAIN95
ANY ERRORS IN RELATED TERMS
NO
WRITE OUTPUT RECORDS ON TAPE
IS CARD TYPE BLANK
NO
E3
YES
NO
YES
MESSPRT
PRINT ERROR MESSAGES
ANY RELATED TERMS NOT FOUND
YES
E119

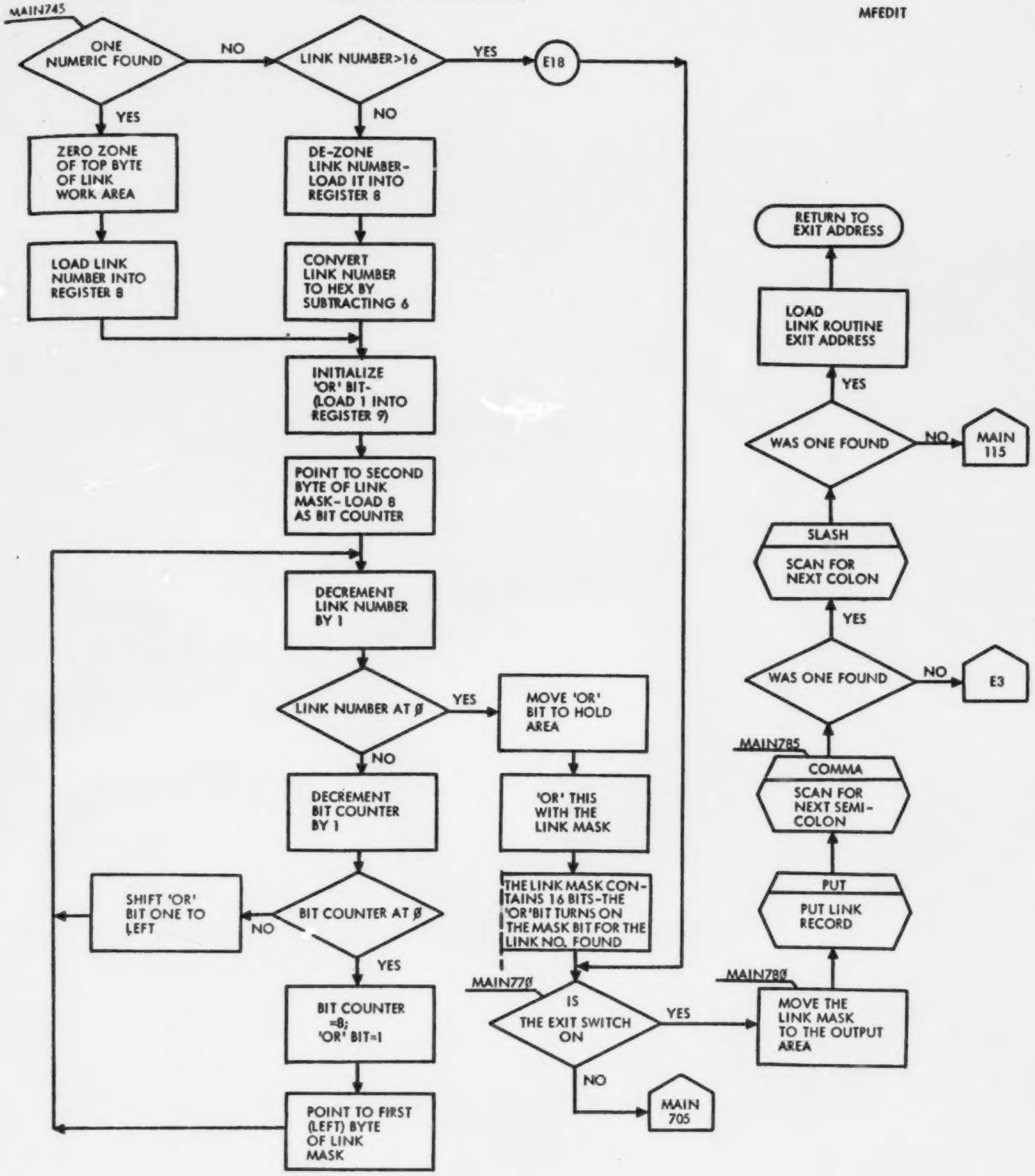

E4
SLASH
SCAN FOR NEXT COLON
WAS ONE FOUND
NO
E11
E3
YES
IS TEXT SWITCH ON
YES
IS DATA TYPE=T
YES
MOVE SEGMENT NUMBER INTO TYPE HOLD AREA
NO
E12
MAIN 9
IS REST OF CARD (AFTER SEQ. NO.) BLANK
YES
E20
NO
E69
NO
MAIN405
IS DATA TYPE=D
NO
YES
TURN OFF CATEGORY AND LINK SWITCHES
IS DATA TYPE=DNNN
YES
VALIDNO
VALIDATE RANK NUMBER
NO
ADD DESCRIPTOR CODE (X'01') TO OUTPUT; INCREMENT PTR.
WAS THERE AN ERROR IN THE NO.
YES
NO
DESCRIPTOR WITH RANK CODE TO OUTPUT (X'07')- BINARY RANK NO. TO OUTPUT
INCREMENT OUTPUT POINTER
MAIN420
COMMA
SCAN FOR NEXT SEMI-COLON
SAVE E49 AS ADDRESS FOR VALIDITY ERROR EXIT
WAS A SEMI-COLON FOUND
NO
TURN SW1 ON
YES
VALIDITY
CHECK VALIDITY OF DESCRIPTOR
PUT
PUT DESCRIPTOR RECORD
TURN DESCRIPTOR SWITCH ON
E49
SAVE E3 AS EXIT FROM CATEGORY ROUTINE
IS SW1 ON
YES
TURN SW1 OFF
CAT 40
NO
SLASH
SCAN FOR NEXT COLON
WAS ONE FOUND
NO
E121
CAT 40
YES
SAVE MAIN 440 AS RETURN ADDRESS FROM CATEGORY
IS DATA TYPE=C
NO
CAT 40
YES
CAT 05


MAIN440
IS DATA TYPE=S
YES
MOVE SUBDESCRIPTOR TYPE CODE TO OUTPUT (X'03')
NO
INCREMENT OUTPUT PTR-TURN LINK SWITCH ON
MAIN 420
IS DATA TYPE A VALUE
YES
SAVE MAIN 440 AS RETURN ADDRESS FROM VALUE SCAN
NO
TURN ON LINK AND CATEGORY SWITCHES
IS DATA TYPE=LK
NO
E8
YES
MOVE LINK TYPE CODE TO OUTPUT (X'05')
IS LINK SWITCH ON
YES
E13
NO
INCREMENT OUTPUT POINTER
TURN LINK SWITCH ON
VSCAN
LINK TYPE CODE (X'05') TO OUTPUT
INCREMENT OUTPUT POINTER
SAVE MAIN 440 AS ADDRESS FOR RETURN FROM LINK ROUT
E7

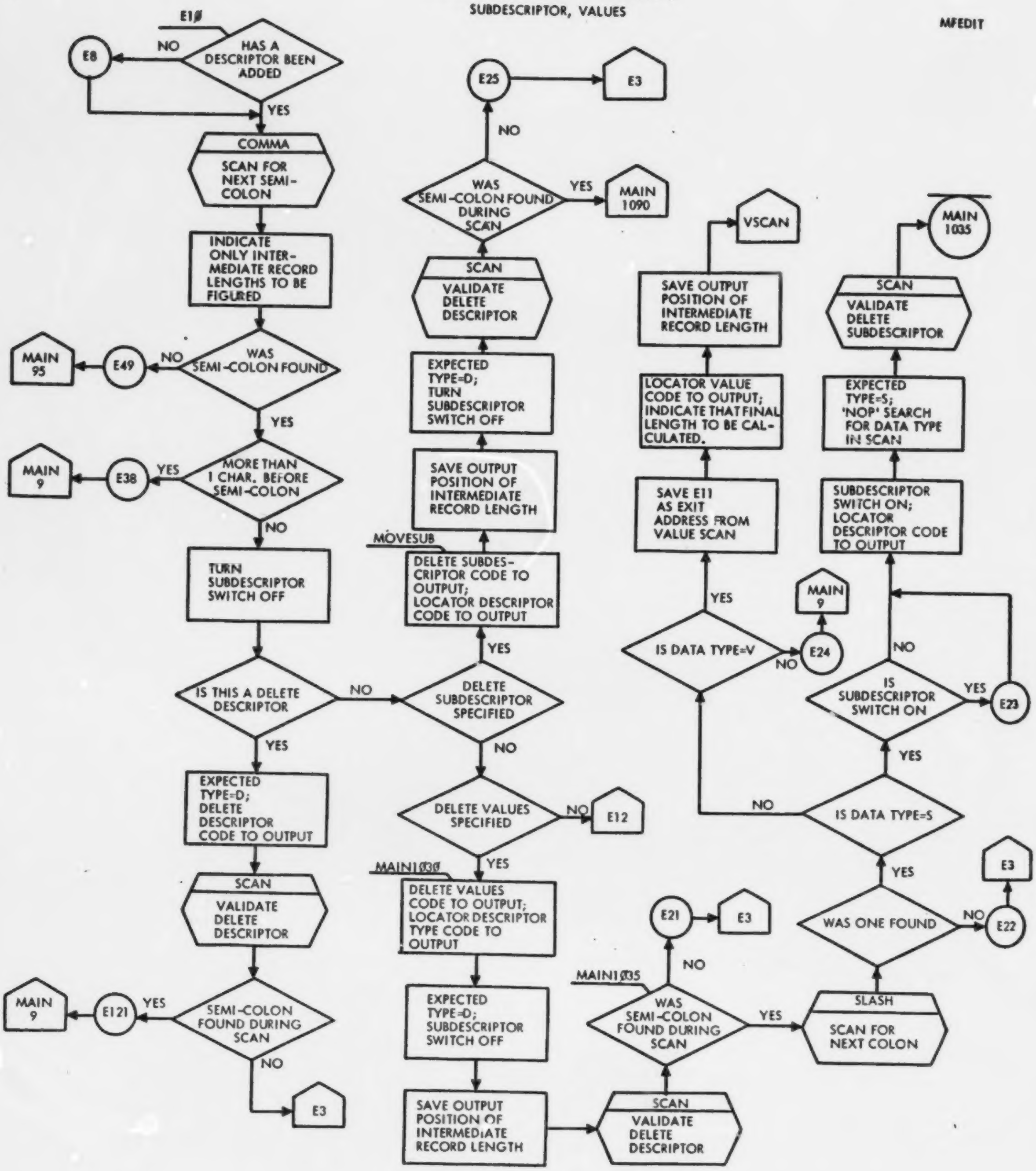
CHECK LINK VALUES


E7
POINT TO END OF INPUT BUFFER
CLEAR LINK MASK- TURN OFF EXIT SWITCH
INCREMENT INPUT BUFFER POINTER BY 1
AT END OF INPUT
NO
YES
READ
READ NEXT INPUT RECORD
IS CARD TYPE BLANK
YES
NO
E14
E3
AT A "("
NO
AT A BLANK
YES
NO
E15
YES
MAIN705
ZERO LINK WORK AREA AND COUNTER
MAIN710
INCREMENT INPUT BUFFER POINTER BY 1
AT END OF BUFFER
YES
READ
READ NEXT INPUT RECORD
IS CARD TYPE BLANK
YES
NO
E14
E3
NO
BUFFER CHARACTER< 0
YES
BUFFER CHARACTER=","
YES
IS DIGIT SW ON
NO
E20
MAIN 780
YES
TURN DIGIT SWITCH OFF
IS LINK WORK AREA STILL 0
NO
MAIN 745
YES
E17
MAIN 770
NO
BUFFER CHARACTER A BLANK
NO
MAIN 19
YES
IS DIGIT SW ON
NO
MAIN 710
YES
E113
MAIN 705
NO
BUFFER CHARACTER>9
YES
E19
MAIN19
MAIN 705
NO
TWO DIGITS MOVED ALREADY
YES
E16
MAIN 785
NO
TURN DIGIT SWITCH ON
MOVE NUMBER TO LINK WORK AREA
INCREMENT COUNTER AND WORK AREA POINTER
MAIN 710

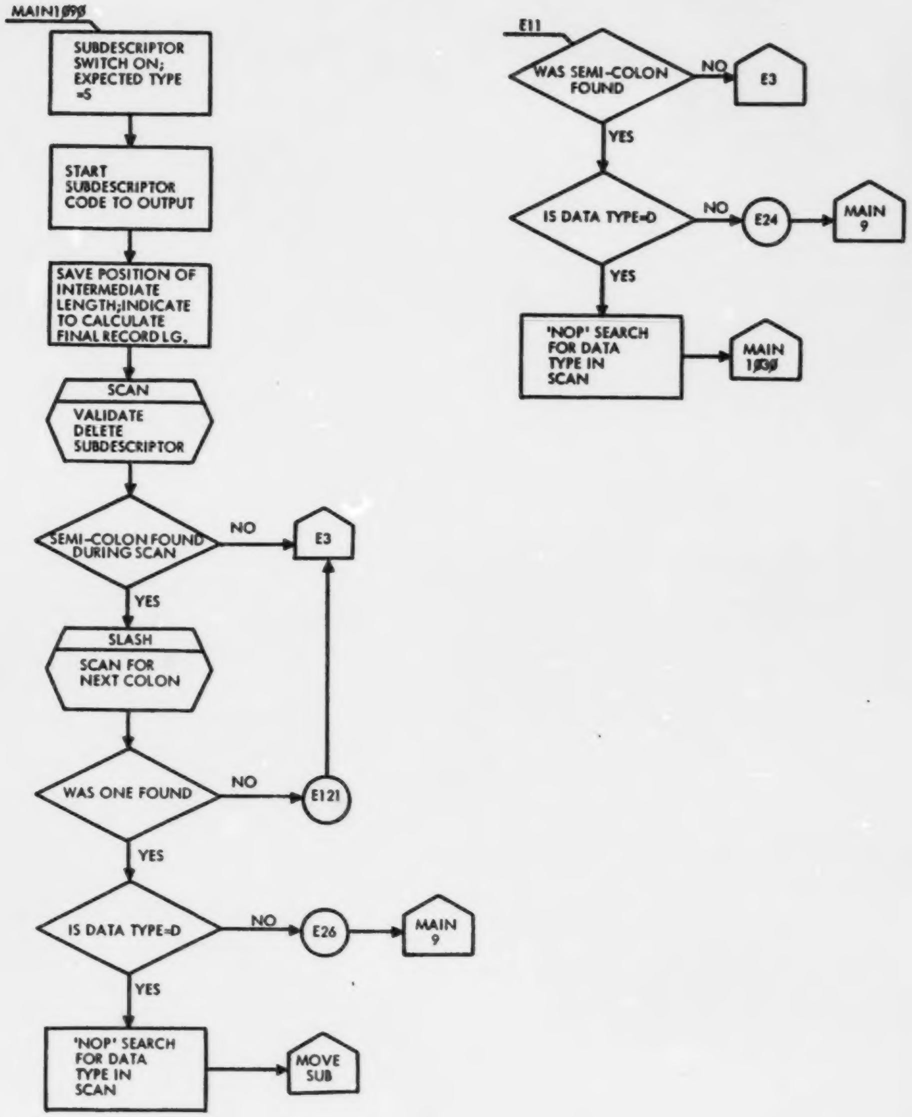
CHECK LINK VALUES (CONT.)


MAIN745
ONE NUMERIC FOUND
NO
LINK NUMBER>16
YES
E18
YES
NO
ZERO ZONE OF TOP BYTE OF LINK WORK AREA
DE-ZONE LINK NUMBER-LOAD IT INTO REGISTER 8
LOAD LINK NUMBER INTO REGISTER 8
CONVERT LINK NUMBER TO HEX BY SUBTRACTING 6
INITIALIZE 'OR' BIT-(LOAD 1 INTO REGISTER 9)
POINT TO SECOND BYTE OF LINK MASK-LOAD 8 AS BIT COUNTER
DECREMENT LINK NUMBER BY 1
LINK NUMBER AT 0
YES
MOVE 'OR' BIT TO HOLD AREA
NO
DECREMENT BIT COUNTER BY 1
'OR' THIS WITH THE LINK MASK
THE LINK MASK CONTAINS 16 BITS-THE 'OR'BIT TURNS ON THE MASK BIT FOR THE LINK NO. FOUND
SHIFT 'OR' BIT ONE TO LEFT
NO
BIT COUNTER AT 0
YES
BIT COUNTER =8; 'OR' BIT=1
POINT TO FIRST (LEFT) BYTE OF LINK MASK
MAIN770
IS THE EXIT SWITCH ON
YES
NO
MAIN 705
MAIN780
MOVE THE LINK MASK TO THE OUTPUT AREA
PUT
PUT LINK RECORD
MAIN785
COMMA
SCAN FOR NEXT SEMI-COLON
WAS ONE FOUND
NO
E3
YES
SLASH
SCAN FOR NEXT COLON
WAS ONE FOUND
NO
MAIN 115
YES
LOAD LINK ROUTINE EXIT ADDRESS
RETURN TO EXIT ADDRESS

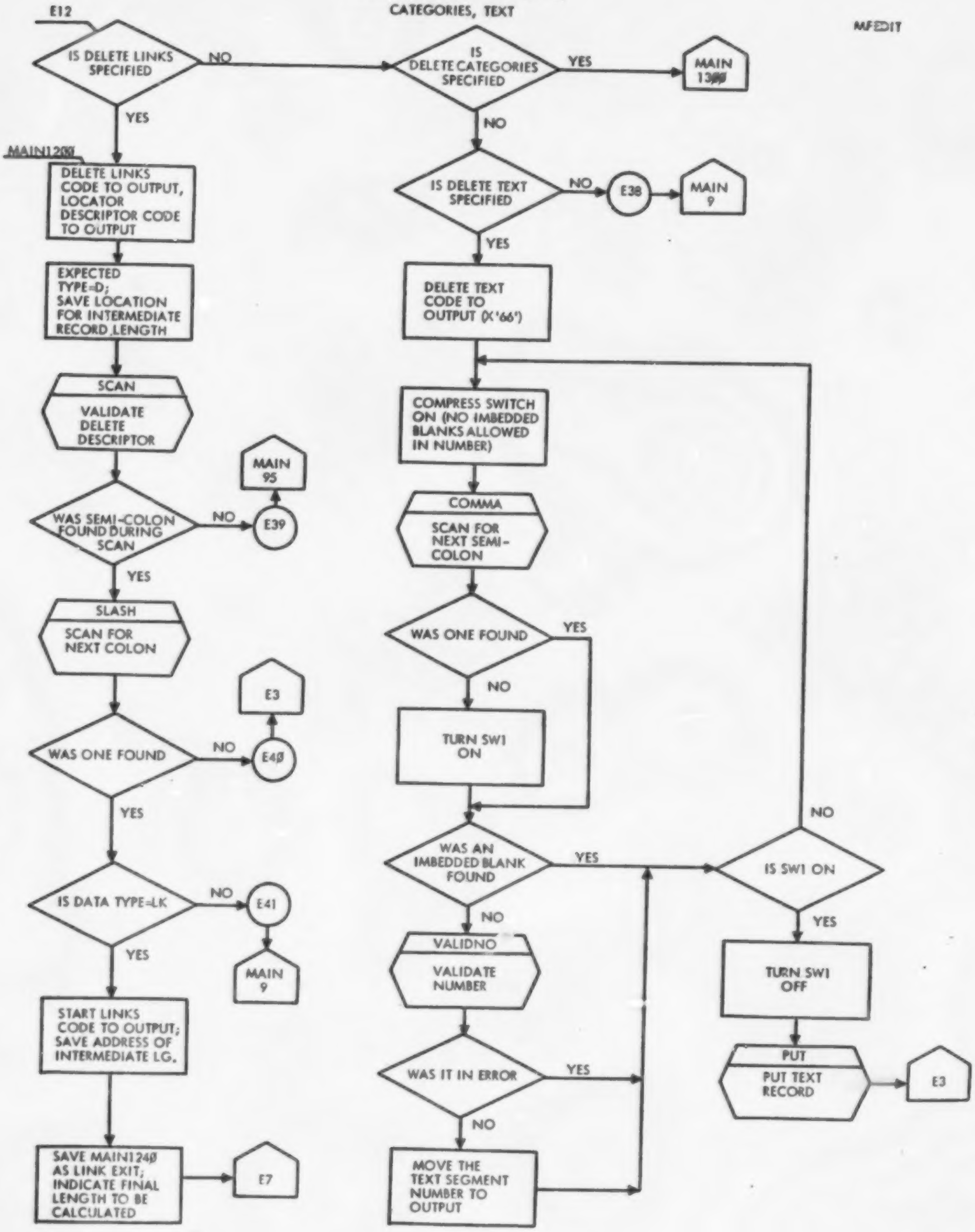
PROCESS DELETE RECORDS- DESCRIPTOR, SUBDESCRIPTOR, VALUES


E10
HAS A DESCRIPTOR BEEN ADDED
NO
E8
YES
COMMA
SCAN FOR NEXT SEMI-COLON
INDICATE ONLY INTER-MEDIATE RECORD LENGTHS TO BE FIGURED
WAS SEMI-COLON FOUND
NO
E49
MAIN 95
YES
MORE THAN 1 CHAR. BEFORE SEMI-COLON
YES
E38
MAIN 9
NO
TURN SUBDESCRIPTOR SWITCH OFF
IS THIS A DELETE DESCRIPTOR
NO
YES
EXPECTED TYPE=D; DELETE DESCRIPTOR CODE TO OUTPUT
SCAN
VALIDATE DELETE DESCRIPTOR
SEMI-COLON FOUND DURING SCAN
YES
E121
MAIN 9
NO
E3
E25
E3
NO
WAS SEMI-COLON FOUND DURING SCAN
YES
MAIN 1090
SCAN
VALIDATE DELETE DESCRIPTOR
EXPECTED TYPE=D; TURN SUBDESCRIPTOR SWITCH OFF
SAVE OUTPUT POSITION OF INTERMEDIATE RECORD LENGTH
MOVESUB
DELETE SUBDES-CRIPTOR CODE TO OUTPUT; LOCATOR DESCRIPTOR CODE TO OUTPUT
YES
DELETE SUBDESCRIPTOR SPECIFIED
NO
DELETE VALUES SPECIFIED
NO
E12
YES
MAIN1030
DELETE VALUES CODE TO OUTPUT; LOCATOR DESCRIPTOR TYPE CODE TO OUTPUT
EXPECTED TYPE=D; SUBDESCRIPTOR SWITCH OFF
SAVE OUTPUT POSITION OF INTERMEDIATE RECORD LENGTH
SCAN
VALIDATE DELETE DESCRIPTOR
MAIN1035
WAS SEMI-COLON FOUND DURING SCAN
NO
E21
E3
YES
SLASH
SCAN FOR NEXT COLON
WAS ONE FOUND
NO
E22
E3
YES
IS DATA TYPE=S
NO
YES
IS SUBDESCRIPTOR SWITCH ON
YES
E23
NO
SUBDESCRIPTOR SWITCH ON; LOCATOR DESCRIPTOR CODE TO OUTPUT
EXPECTED TYPE=S; 'NOP' SEARCH FOR DATA TYPE IN SCAN
SCAN
VALIDATE DELETE SUBDESCRIPTOR
MAIN 1035
IS DATA TYPE=V
NO
E24
MAIN 9
YES
SAVE E11 AS EXIT ADDRESS FROM VALUE SCAN
LOCATOR VALUE CODE TO OUTPUT; INDICATE THAT FINAL LENGTH TO BE CAL-CULATED.
SAVE OUTPUT POSITION OF INTERMEDIATE RECORD LENGTH
VSCAN

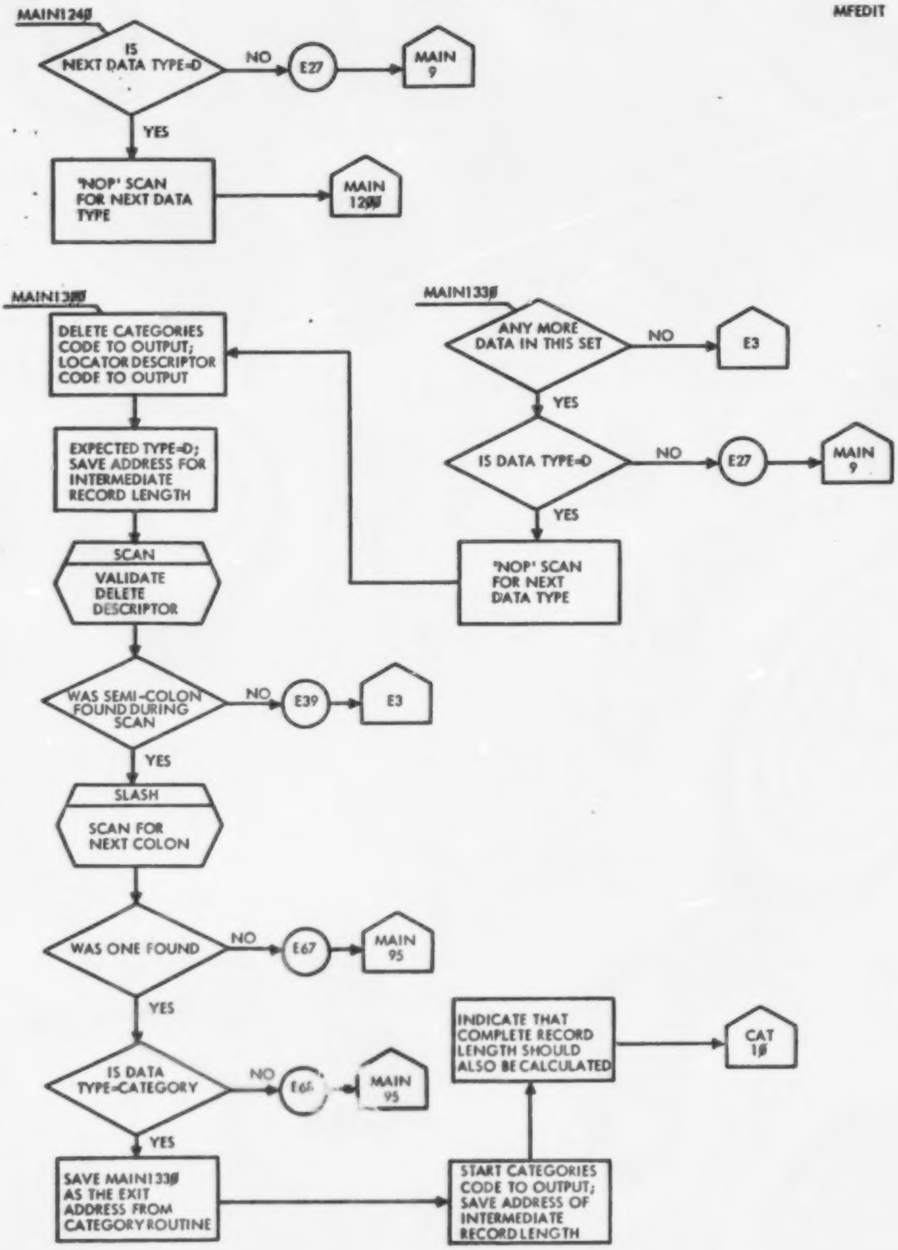
PROCESS DELETE RECORDS- DESCRIPTOR, SUBDESCRIPTOR, VALUES (CONT)


MAIN1090
SUBDESCRIPTOR SWITCH ON; EXPECTED TYPE =S
START SUBDESCRIPTOR CODE TO OUTPUT
SAVE POSITION OF INTERMEDIATE LENGTH;INDICATE TO CALCULATE FINAL RECORD LG.
SCAN
VALIDATE DELETE SUBDESCRIPTOR
SEMI-COLON FOUND DURING SCAN
NO
E3
YES
SLASH
SCAN FOR NEXT COLON
WAS ONE FOUND
NO
E121
YES
IS DATA TYPE=D
NO
E26
MAIN 9
YES
'NOP' SEARCH FOR DATA TYPE IN SCAN
MOVE SUB
E11
WAS SEMI-COLON FOUND
NO
E3
YES
IS DATA TYPE=D
NO
E24
MAIN 9
YES
'NOP' SEARCH FOR DATA TYPE IN SCAN
MAIN 1030

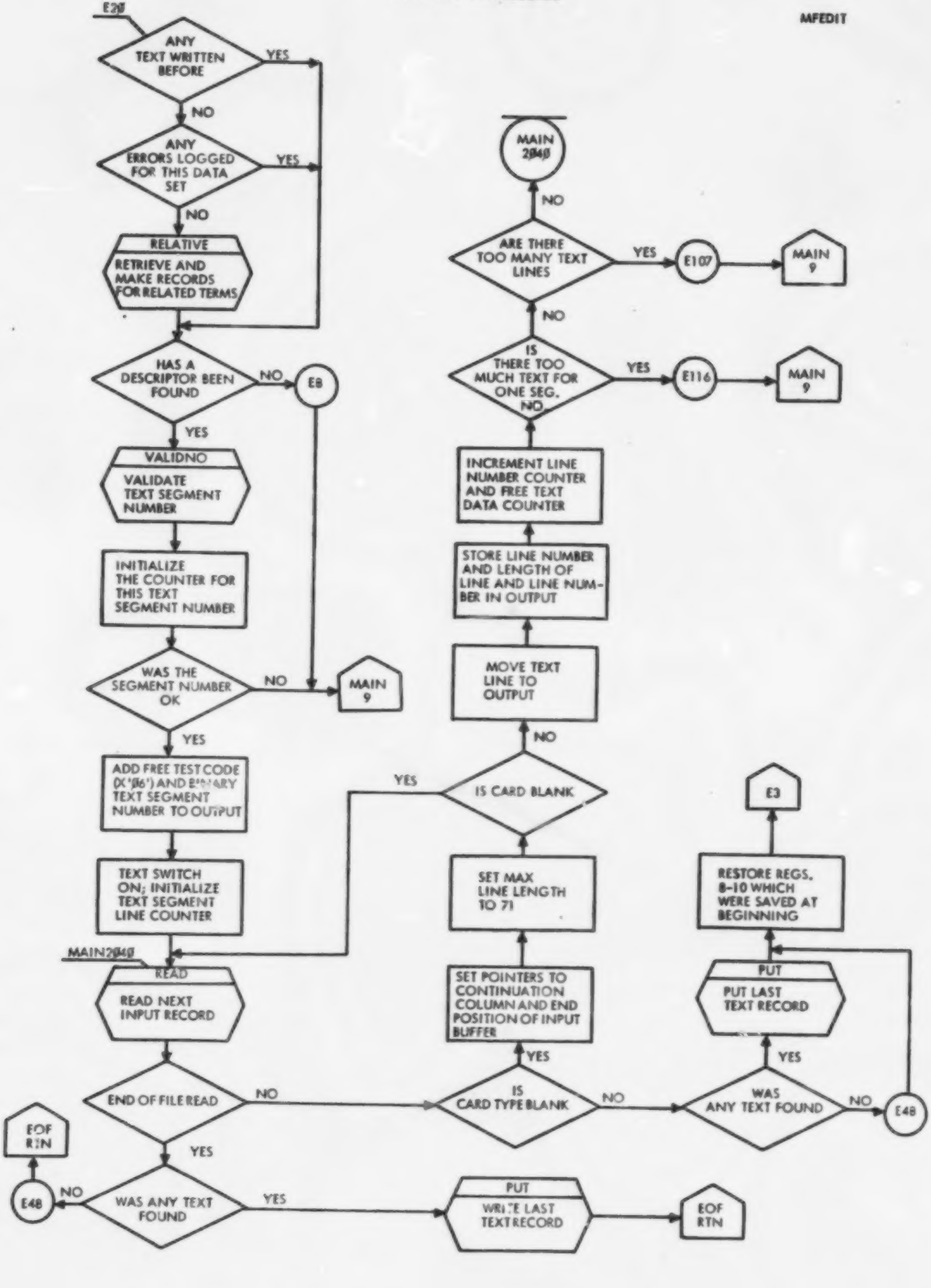
PROCESS DELETE RECORDS- LINKS, CATEGORIES, TEXT


E12
IS DELETE LINKS SPECIFIED
NO
IS DELETE CATEGORIES SPECIFIED
YES
MAIN 1300
YES
NO
MAIN1200
DELETE LINKS CODE TO OUTPUT, LOCATOR DESCRIPTOR CODE TO OUTPUT
IS DELETE TEXT SPECIFIED
NO
E38
MAIN 9
YES
EXPECTED TYPE=D; SAVE LOCATION FOR INTERMEDIATE RECORD LENGTH
DELETE TEXT CODE TO OUTPUT (X'66')
SCAN
VALIDATE DELETE DESCRIPTOR
COMPRESS SWITCH ON (NO IMBEDDED BLANKS ALLOWED IN NUMBER)
MAIN 95
WAS SEMI-COLON FOUND DURING SCAN
NO
E39
COMMA
SCAN FOR NEXT SEMI-COLON
YES
SLASH
SCAN FOR NEXT COLON
WAS ONE FOUND
YES
NO
E3
TURN SW1 ON
WAS ONE FOUND
NO
E40
YES
NO
WAS AN IMBEDDED BLANK FOUND
YES
IS SW1 ON
NO
IS DATA TYPE=LK
NO
E41
VALIDNO
VALIDATE NUMBER
YES
YES
MAIN 9
TURN SW1 OFF
START LINKS CODE TO OUTPUT; SAVE ADDRESS OF INTERMEDIATE LG.
WAS IT IN ERROR
YES
PUT
PUT TEXT RECORD
E3
NO
SAVE MAIN1240 AS LINK EXIT; INDICATE FINAL LENGTH TO BE CALCULATED
E7
MOVE THE TEXT SEGMENT NUMBER TO OUTPUT

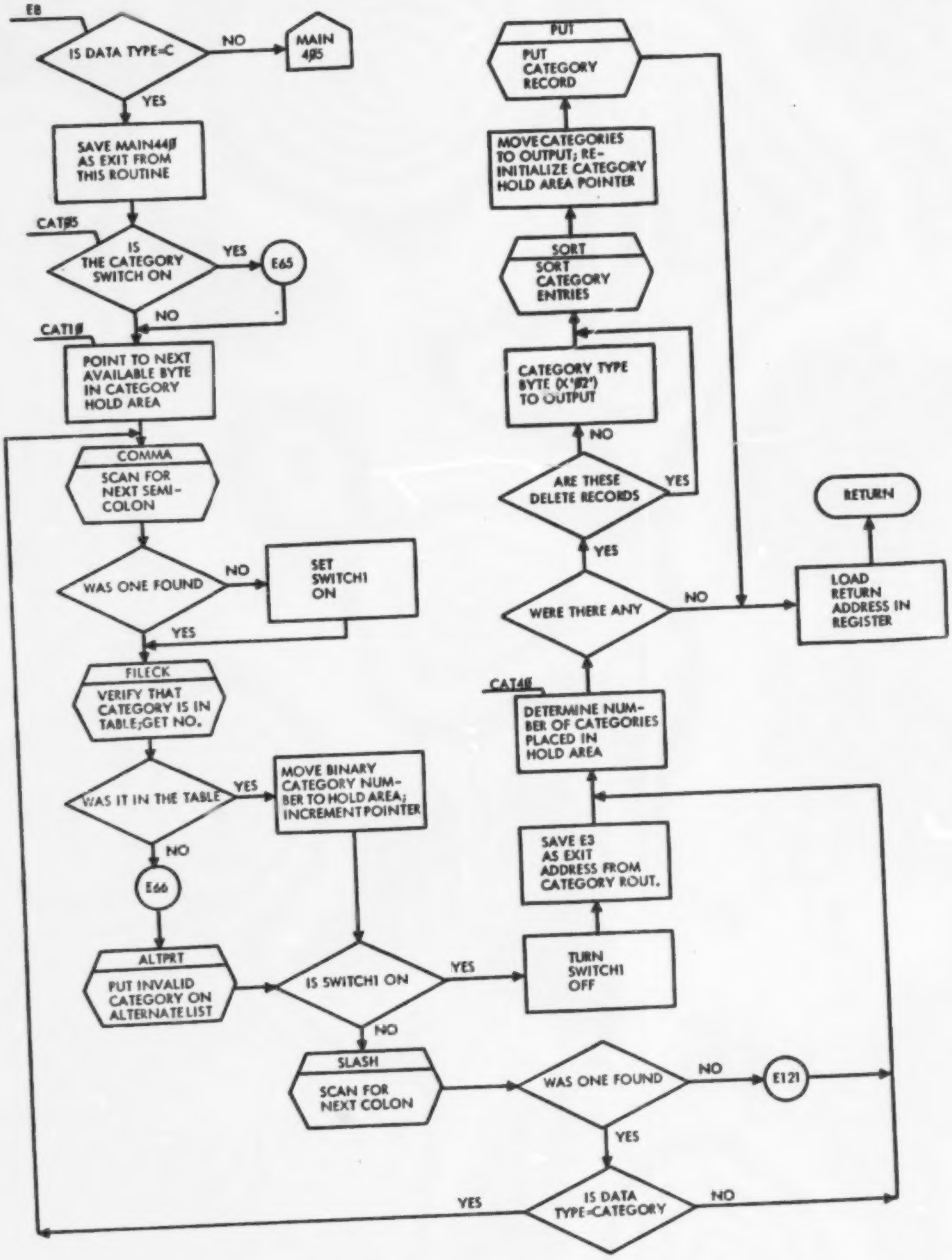
PROCESS DELETE RECORDS- LINKS CATEGORIES, TEXT (CONT.)


MAIN1240
IS NEXT DATA TYPE=D
NO
E27
MAIN 9
YES
'NOP' SCAN FOR NEXT DATA TYPE
MAIN 1290
MAIN1300
DELETE CATEGORIES CODE TO OUTPUT; LOCATOR DESCRIPTOR CODE TO OUTPUT
EXPECTED TYPE=D; SAVE ADDRESS FOR INTERMEDIATE RECORD LENGTH
SCAN
VALIDATE DELETE DESCRIPTOR
WAS SEMI-COLON FOUND DURING SCAN
NO
E39
E3
YES
SLASH
SCAN FOR NEXT COLON
WAS ONE FOUND
NO
E67
MAIN 95
YES
IS DATA TYPE=CATEGORY
NO
E6
MAIN 95
YES
SAVE MAIN1330 AS THE EXIT ADDRESS FROM CATEGORY ROUTINE
START CATEGORIES CODE TO OUTPUT; SAVE ADDRESS OF INTERMEDIATE RECORD LENGTH
INDICATE THAT COMPLETE RECORD LENGTH SHOULD ALSO BE CALCULATED
CAT 10
MAIN1330
ANY MORE DATA IN THIS SET
NO
E3
YES
IS DATA TYPE=D
NO
E27
MAIN 9
YES
'NOP' SCAN FOR NEXT DATA TYPE

MAKE ADD TEXT RECORDS

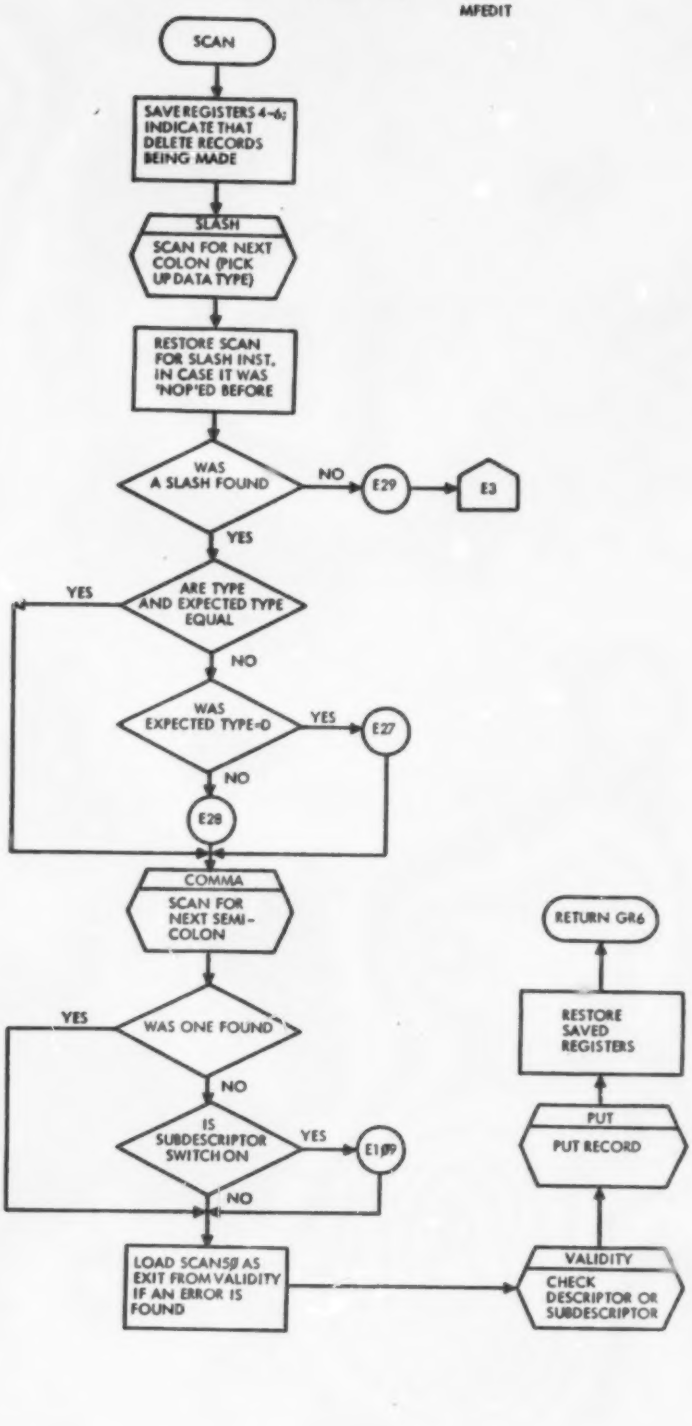

E20
ANY TEXT WRITTEN BEFORE
YES
NO
ANY ERRORS LOGGED FOR THIS DATA SET
YES
NO
RELATIVE
RETRIEVE AND MAKE RECORDS FOR RELATED TERMS
HAS A DESCRIPTOR BEEN FOUND
NO
E8
YES
VALIDNO
VALIDATE TEXT SEGMENT NUMBER
INITIALIZE THE COUNTER FOR THIS TEXT SEGMENT NUMBER
WAS THE SEGMENT NUMBER OK
NO
MAIN 9
YES
ADD FREE TEST CODE (X'06') AND BINARY TEXT SEGMENT NUMBER TO OUTPUT
TEXT SWITCH ON; INITIALIZE TEXT SEGMENT LINE COUNTER
MAIN2040
READ
READ NEXT INPUT RECORD
END OF FILE READ
NO
YES
WAS ANY TEXT FOUND
NO
E48
EOF RTN
YES
PUT
WRITE LAST TEXT RECORD
EOF RTN
IS CARD TYPE BLANK
YES
SET POINTERS TO CONTINUATION COLUMN AND END POSITION OF INPUT BUFFER
SET MAX LINE LENGTH TO 71
IS CARD BLANK
YES
NO
MOVE TEXT LINE TO OUTPUT
STORE LINE NUMBER AND LENGTH OF LINE AND LINE NUMBER IN OUTPUT
INCREMENT LINE NUMBER COUNTER AND FREE TEXT DATA COUNTER
IS THERE TOO MUCH TEXT FOR ONE SEG.
YES
E116
MAIN 9
NO
ARE THERE TOO MANY TEXT LINES
YES
E107
MAIN 9
NO
MAIN 2040
NO
WAS ANY TEXT FOUND
YES
NO
E48
PUT
PUT LAST TEXT RECORD
RESTORE REGS. 8-10 WHICH WERE SAVED AT BEGINNING
E3

PROCESS CATEGORY ENTRIES

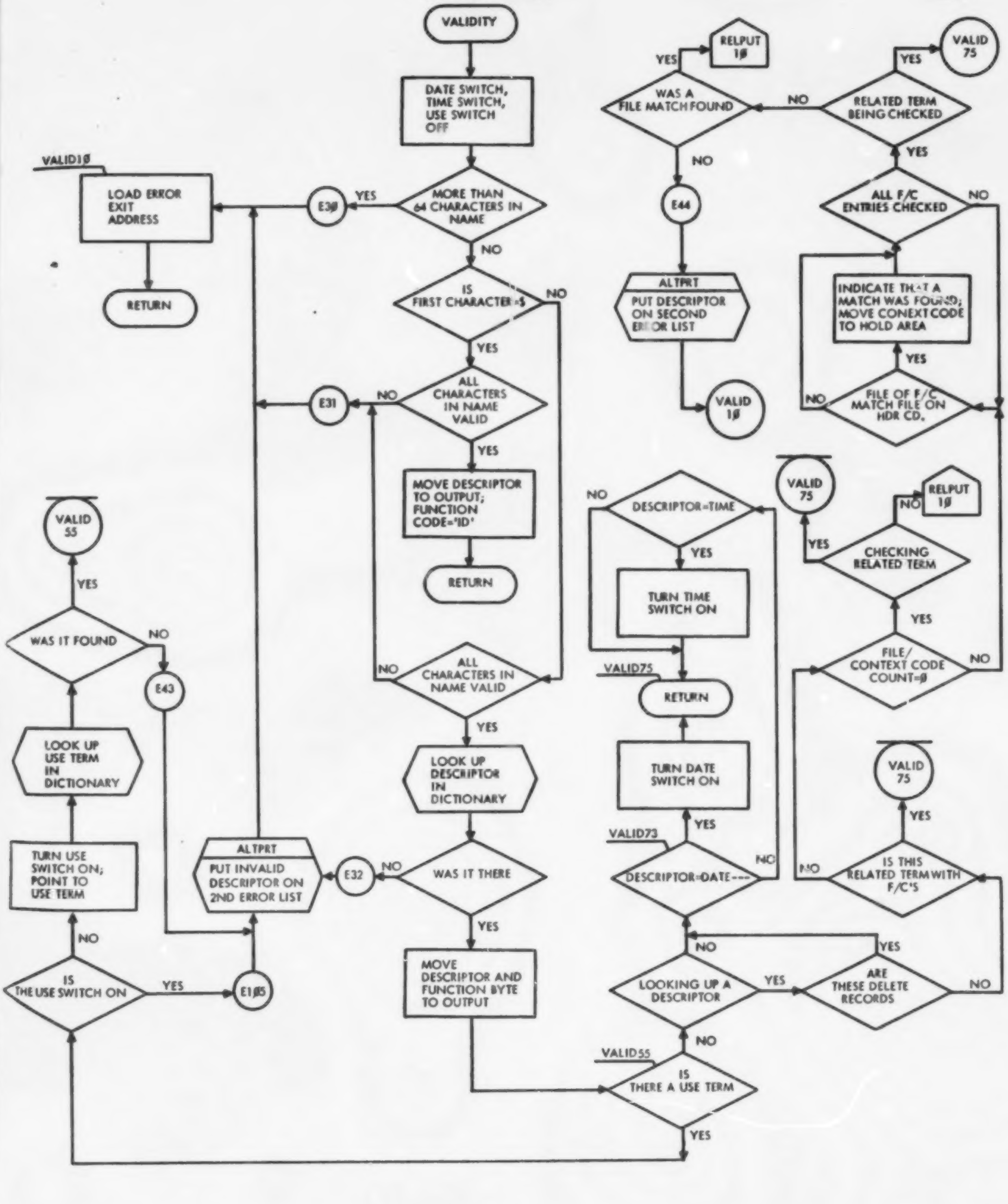

E8
IS DATA TYPE=C
NO
MAIN 405
YES
SAVE MAIN440 AS EXIT FROM THIS ROUTINE
CAT05
IS THE CATEGORY SWITCH ON
YES
E65
NO
CAT10
POINT TO NEXT AVAILABLE BYTE IN CATEGORY HOLD AREA
COMMA
SCAN FOR NEXT SEMI-COLON
WAS ONE FOUND
NO
SET SWITCH1 ON
YES
FILECK
VERIFY THAT CATEGORY IS IN TABLE;GET NO.
WAS IT IN THE TABLE
YES
MOVE BINARY CATEGORY NUMBER TO HOLD AREA; INCREMENT POINTER
NO
E66
ALTPRT
PUT INVALID CATEGORY ON ALTERNATE LIST
IS SWITCH1 ON
YES
TURN SWITCH1 OFF
NO
SLASH
SCAN FOR NEXT COLON
WAS ONE FOUND
NO
E121
YES
IS DATA TYPE=CATEGORY
YES
NO
SAVE E3 AS EXIT ADDRESS FROM CATEGORY ROUT.
CAT40
DETERMINE NUMBER OF CATEGORIES PLACED IN HOLD AREA
WERE THERE ANY
NO
YES
ARE THESE DELETE RECORDS
YES
NO
CATEGORY TYPE BYTE (X'02') TO OUTPUT
SORT
SORT CATEGORY ENTRIES
MOVE CATEGORIES TO OUTPUT; RE-INITIALIZE CATEGORY HOLD AREA POINTER
PUT
PUT CATEGORY RECORD
LOAD RETURN ADDRESS IN REGISTER
RETURN

DESCRIPTOR - SUBDESCRIPTOR SCAN (DELETE RECORDS ONLY)

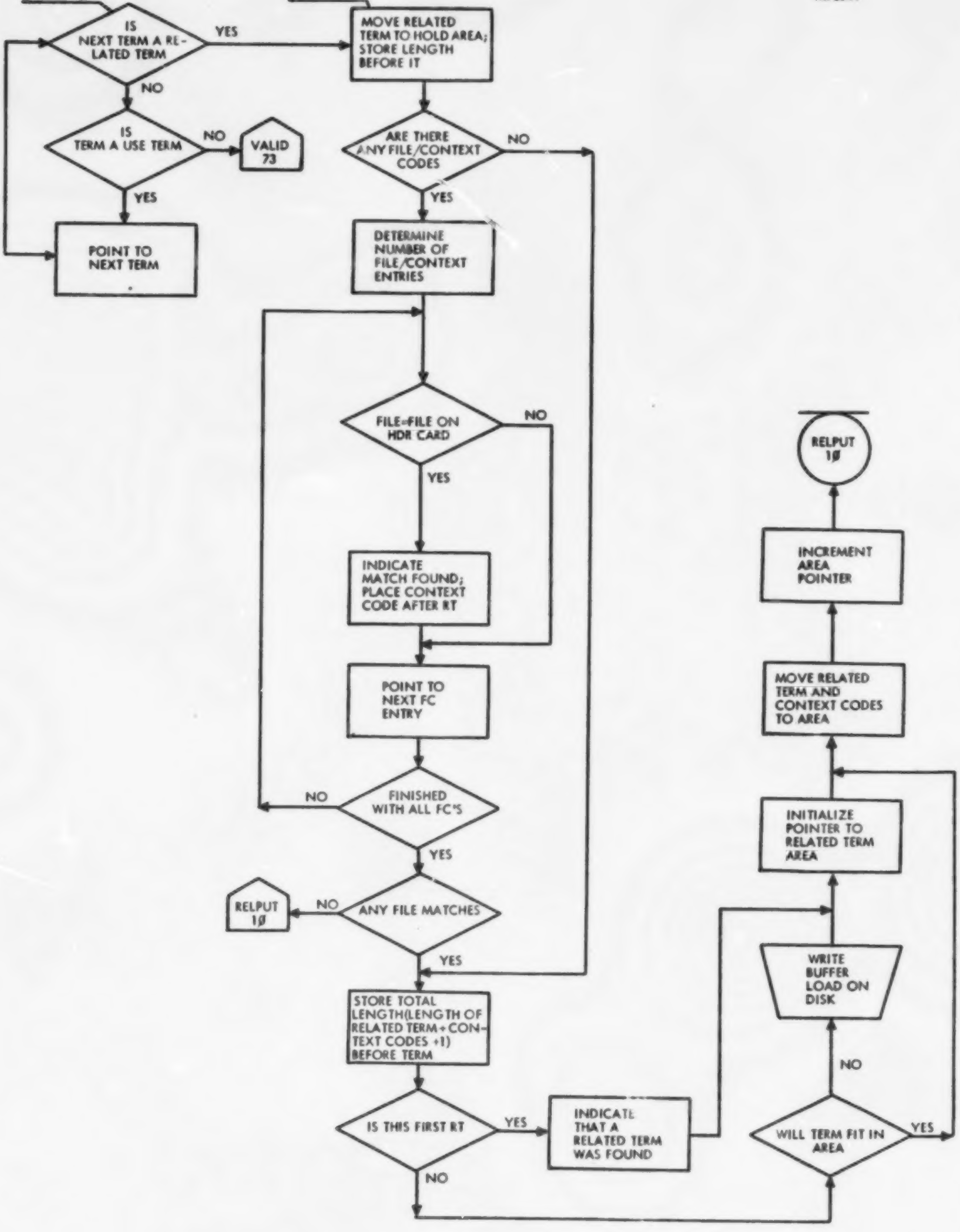
MFEDIT


SCAN
SAVE REGISTERS 4-6; INDICATE THAT DELETE RECORDS BEING MADE
SLASH
SCAN FOR NEXT COLON (PICK UP DATA TYPE)
RESTORE SCAN FOR SLASH INST. IN CASE IT WAS 'NOP'ED BEFORE
WAS A SLASH FOUND
NO
E29
E3
YES
ARE TYPE AND EXPECTED TYPE EQUAL
YES
NO
WAS EXPECTED TYPE=D
YES
E27
NO
E28
COMMA
SCAN FOR NEXT SEMI-COLON
WAS ONE FOUND
YES
NO
IS SUBDESCRIPTOR SWITCH ON
YES
E109
NO
LOAD SCAN50 AS EXIT FROM VALIDITY IF AN ERROR IS FOUND
VALIDITY
CHECK DESCRIPTOR OR SUBDESCRIPTOR
PUT
PUT RECORD
RESTORE SAVED REGISTERS
RETURN GR6

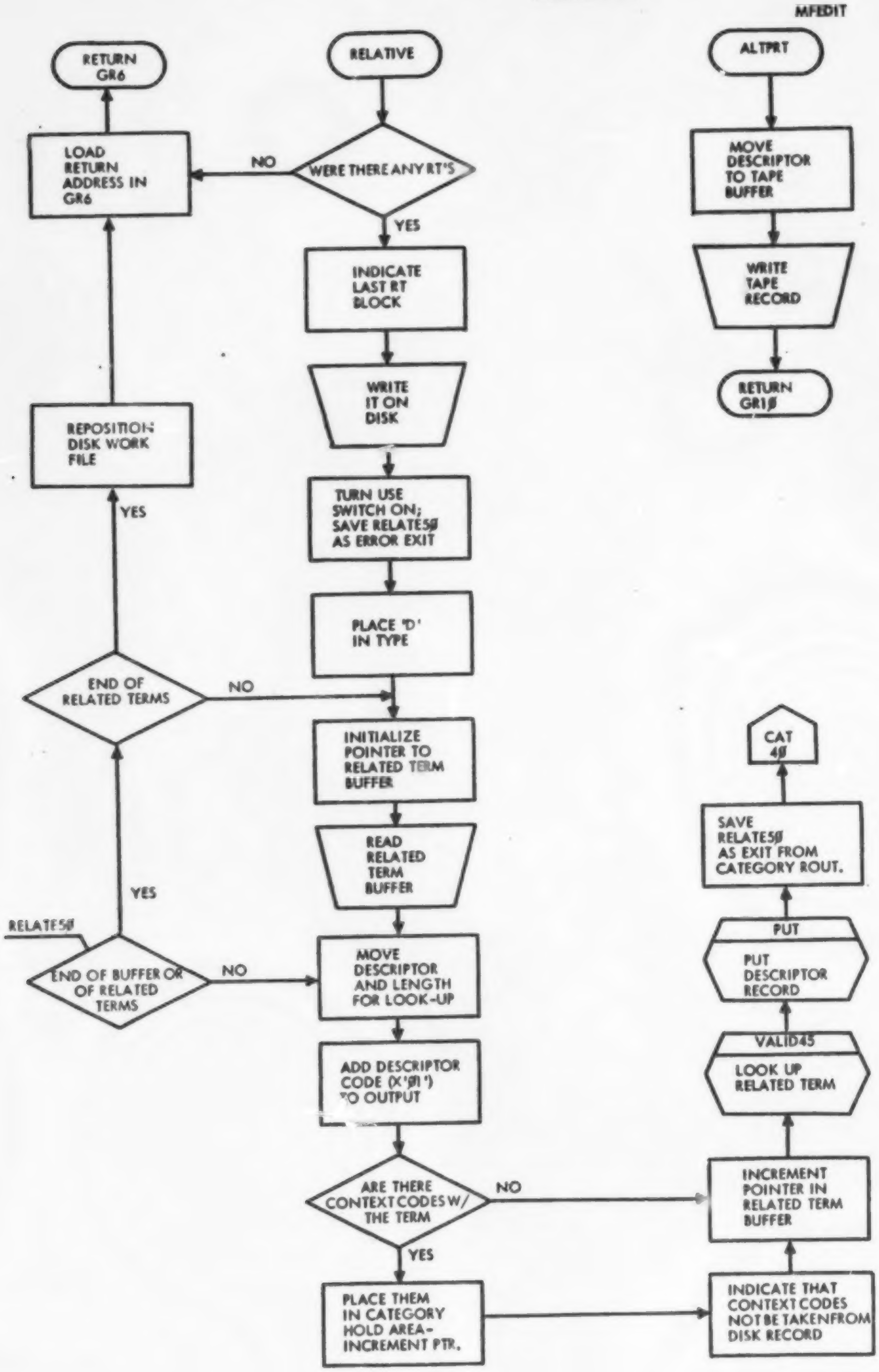
VALIDITY - PROCESS DESCRIPTORS AND SUBDESCRIPTORS


VALIDITY
DATE SWITCH, TIME SWITCH, USE SWITCH OFF
MORE THAN 64 CHARACTERS IN NAME
YES
E30
VALID10
LOAD ERROR EXIT ADDRESS
RETURN
NO
IS FIRST CHARACTER=$
NO
YES
ALL CHARACTERS IN NAME VALID
NO
E31
YES
MOVE DESCRIPTOR TO OUTPUT; FUNCTION CODE='ID'
RETURN
NO
ALL CHARACTERS IN NAME VALID
YES
LOOK UP DESCRIPTOR IN DICTIONARY
WAS IT THERE
NO
E32
ALTPRT
PUT INVALID DESCRIPTOR ON 2ND ERROR LIST
YES
MOVE DESCRIPTOR AND FUNCTION BYTE TO OUTPUT
VALID 55
YES
WAS IT FOUND
NO
E43
LOOK UP USE TERM IN DICTIONARY
TURN USE SWITCH ON; POINT TO USE TERM
NO
IS THE USE SWITCH ON
YES
E105
RELPUT 10
YES
WAS A FILE MATCH FOUND
NO
E44
ALTPRT
PUT DESCRIPTOR ON SECOND ERROR LIST
VALID 10
VALID 75
YES
NO
RELATED TERM BEING CHECKED
YES
ALL F/C ENTRIES CHECKED
NO
INDICATE THAT A MATCH WAS FOUND; MOVE CONEXT CODE TO HOLD AREA
YES
NO
FILE OF F/C MATCH FILE ON HDR CD.
NO
DESCRIPTOR=TIME
YES
TURN TIME SWITCH ON
VALID75
RETURN
TURN DATE SWITCH ON
YES
VALID73
DESCRIPTOR=DATE---
NO
VALID 75
RELPUT 10
NO
YES
CHECKING RELATED TERM
YES
FILE/ CONTEXT CODE COUNT=0
NO
VALID 75
YES
IS THIS RELATED TERM WITH F/C'S
NO
NO
YES
LOOKING UP A DESCRIPTOR
YES
ARE THESE DELETE RECORDS
NO
NO
VALID55
IS THERE A USE TERM
YES

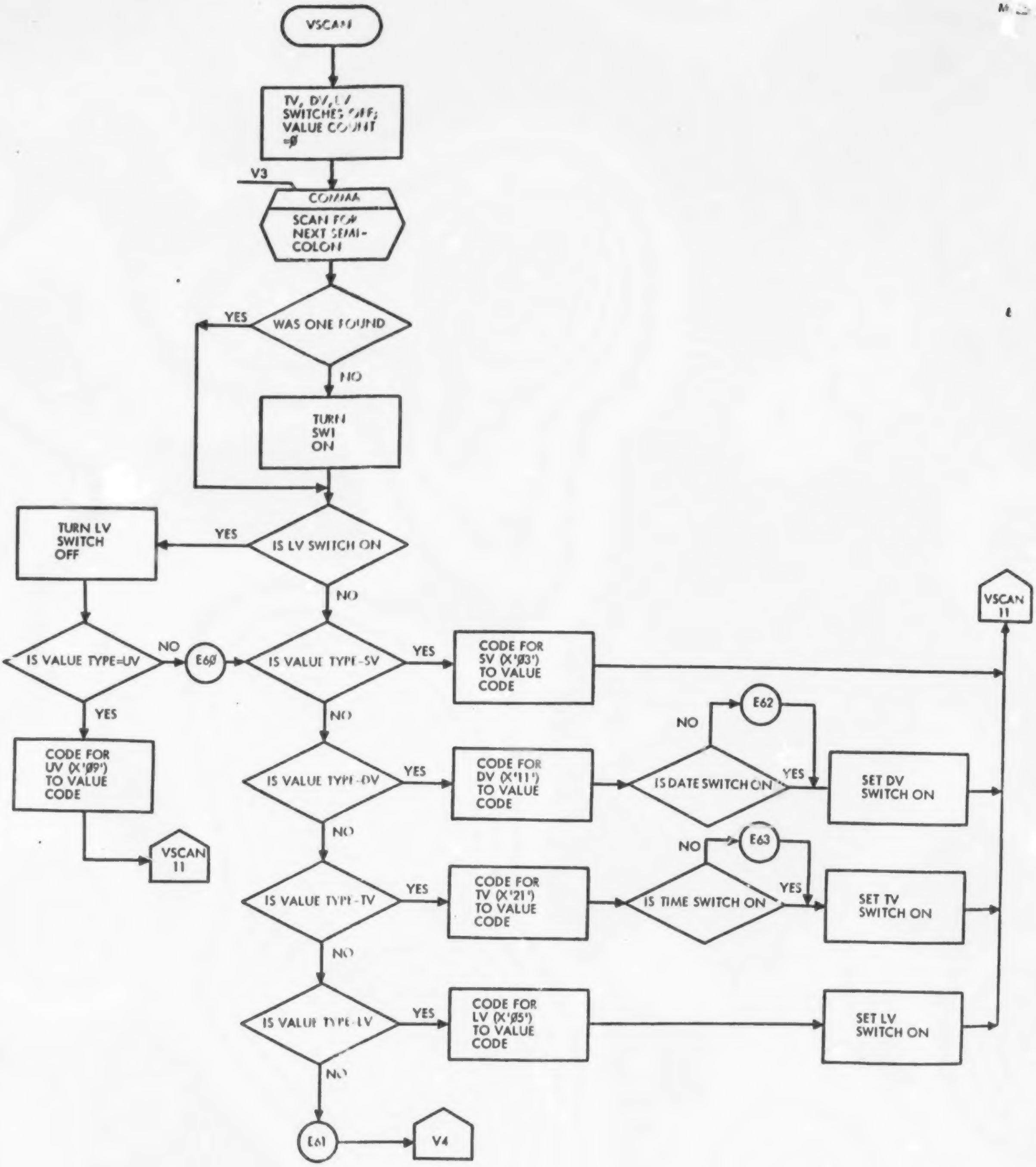
VALIDITY(CONT.) - STORE RELATED TERMS AND THEIR CONTEXT CODES

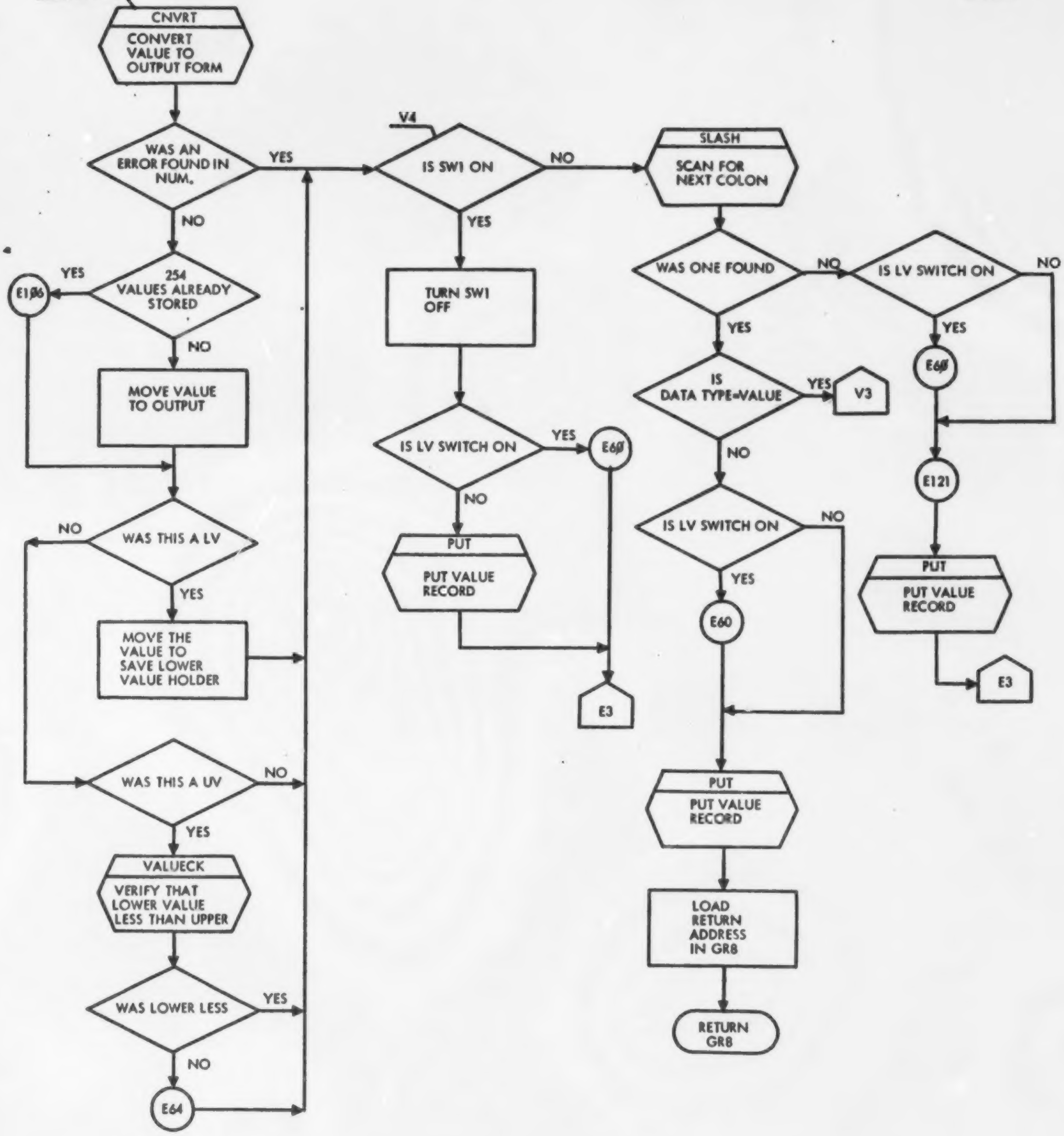

RELPUT10
IS NEXT TERM A RELATED TERM
YES
NO
IS TERM A USE TERM
NO
VALID 73
YES
POINT TO NEXT TERM
RELPUT20
MOVE RELATED TERM TO HOLD AREA; STORE LENGTH BEFORE IT
ARE THERE ANY FILE/CONTEXT CODES
NO
YES
DETERMINE NUMBER OF FILE/CONTEXT ENTRIES
FILE=FILE ON HDR CARD
NO
YES
INDICATE MATCH FOUND; PLACE CONTEXT CODE AFTER RT
POINT TO NEXT FC ENTRY
FINISHED WITH ALL FC'S
NO
YES
ANY FILE MATCHES
NO
RELPUT 10
YES
STORE TOTAL LENGTH(LENGTH OF RELATED TERM + CONTEXT CODES +1) BEFORE TERM
IS THIS FIRST RT
YES
NO
INDICATE THAT A RELATED TERM WAS FOUND
WILL TERM FIT IN AREA
YES
NO
WRITE BUFFER LOAD ON DISK
INITIALIZE POINTER TO RELATED TERM AREA
MOVE RELATED TERM AND CONTEXT CODES TO AREA
INCREMENT AREA POINTER
RELPUT 10

RELATIVE - RETRIEVE RELATED TERMS; PROCESS THEM


MFEDIT
RETURN GR6
LOAD RETURN ADDRESS IN GR6
RELATIVE
WERE THERE ANY RT'S
NO
YES
INDICATE LAST RT BLOCK
WRITE IT ON DISK
TURN USE SWITCH ON; SAVE RELATE50 AS ERROR EXIT
PLACE 'D' IN TYPE
INITIALIZE POINTER TO RELATED TERM BUFFER
READ RELATED TERM BUFFER
MOVE DESCRIPTOR AND LENGTH FOR LOOK-UP
ADD DESCRIPTOR CODE (X'01') TO OUTPUT
ARE THERE CONTEXT CODES W/ THE TERM
NO
YES
PLACE THEM IN CATEGORY HOLD AREA-INCREMENT PTR.
INDICATE THAT CONTEXT CODES NOT BE TAKEN FROM DISK RECORD
INCREMENT POINTER IN RELATED TERM BUFFER
VALID45
LOOK UP RELATED TERM
PUT
PUT DESCRIPTOR RECORD
SAVE RELATE50 AS EXIT FROM CATEGORY ROUT.
CAT 40
REPOSITION DISK WORK FILE
YES
END OF RELATED TERMS
NO
RELATE50
END OF BUFFER OR OF RELATED TERMS
YES
NO
ALTPRT
MOVE DESCRIPTOR TO TAPE BUFFER
WRITE TAPE RECORD
RETURN GR10

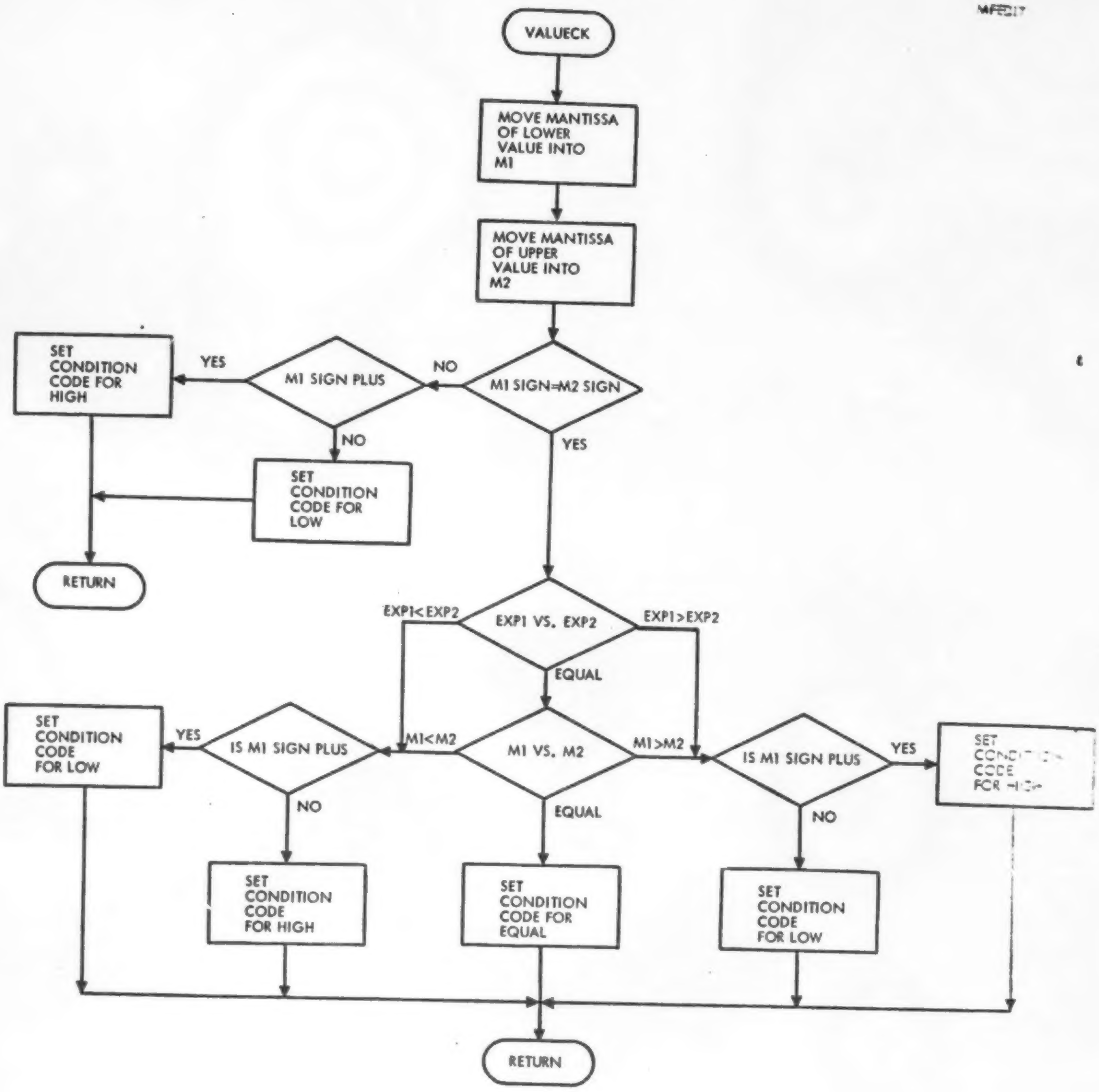
VSCAN- DETERMINES TYPES OF VALUES PRESENT


VSCAN
TV, DV, LV SWITCHES OFF; VALUE COUNT =0
V3
COMMA
SCAN FOR NEXT SEMI-COLON
WAS ONE FOUND
YES
NO
TURN SW1 ON
IS LV SWITCH ON
YES
NO
TURN LV SWITCH OFF
IS VALUE TYPE=UV
NO
YES
CODE FOR UV (X'09') TO VALUE CODE
VSCAN 11
E60
IS VALUE TYPE-SV
YES
NO
CODE FOR SV (X'03') TO VALUE CODE
VSCAN 11
IS VALUE TYPE-DV
YES
NO
CODE FOR DV (X'11') TO VALUE CODE
IS DATE SWITCH ON
NO
E62
YES
SET DV SWITCH ON
IS VALUE TYPE-TV
YES
NO
CODE FOR TV (X'21') TO VALUE CODE
IS TIME SWITCH ON
NO
E63
YES
SET TV SWITCH ON
IS VALUE TYPE-LV
YES
NO
CODE FOR LV (X'05') TO VALUE CODE
SET LV SWITCH ON
E61
V4


VSCAN11
CNVRT
CONVERT VALUE TO OUTPUT FORM
WAS AN ERROR FOUND IN NUM.
YES
NO
254 VALUES ALREADY STORED
YES
E106
NO
MOVE VALUE TO OUTPUT
WAS THIS A LV
NO
YES
MOVE THE VALUE TO SAVE LOWER VALUE HOLDER
WAS THIS A UV
NO
YES
VALUECK
VERIFY THAT LOWER VALUE LESS THAN UPPER
WAS LOWER LESS
YES
NO
E64
V4
IS SW1 ON
NO
YES
TURN SW1 OFF
IS LV SWITCH ON
YES
E60
NO
PUT
PUT VALUE RECORD
E3
SLASH
SCAN FOR NEXT COLON
WAS ONE FOUND
NO
YES
IS DATA TYPE=VALUE
YES
V3
NO
IS LV SWITCH ON
NO
YES
E60
PUT
PUT VALUE RECORD
LOAD RETURN ADDRESS IN GR8
RETURN GR8
IS LV SWITCH ON
NO
YES
E60
E121
PUT
PUT VALUE RECORD
E3

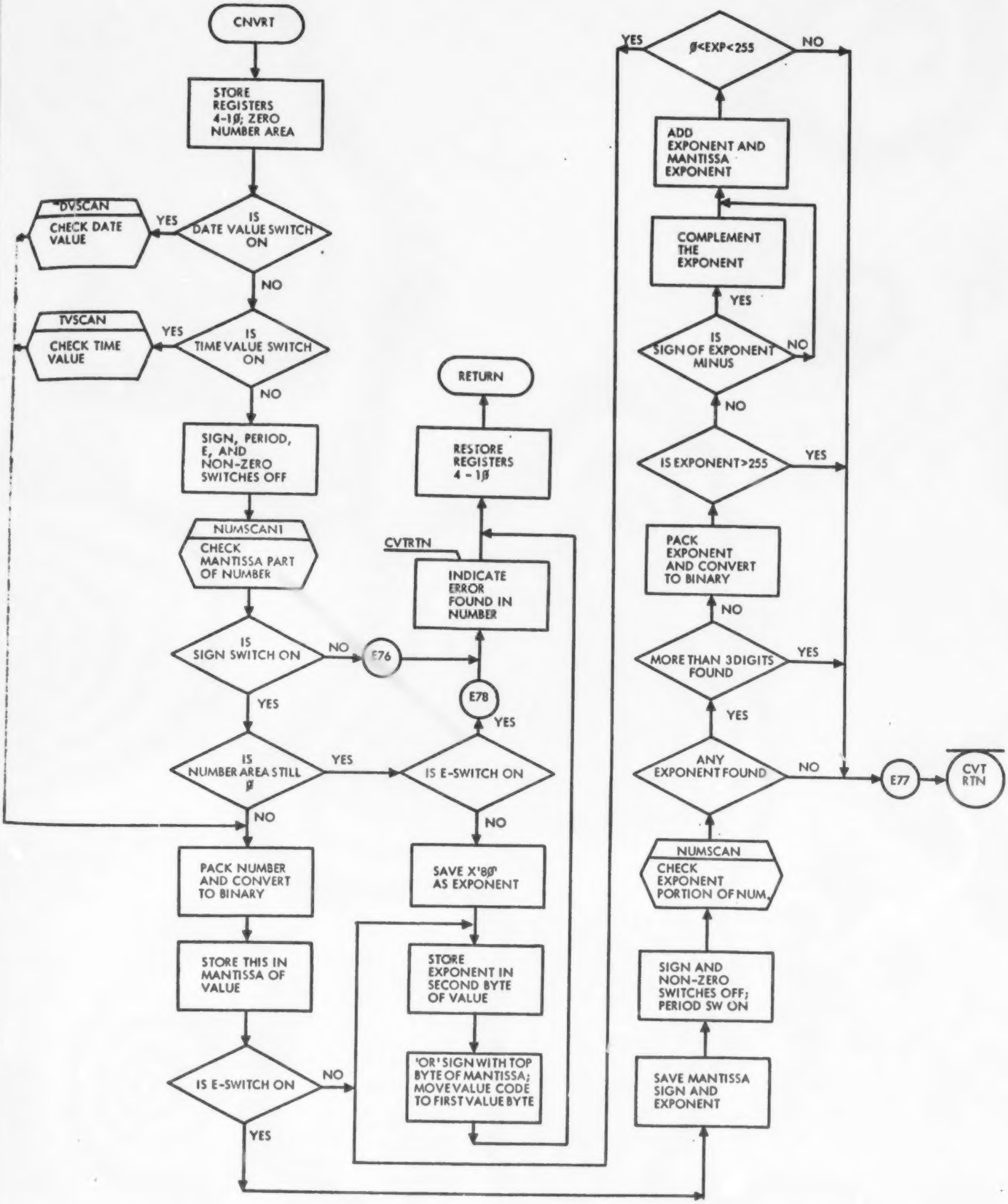
VALUECK- IS LOWER VALUE LESS
THAN UPPER VALUE

MFE017

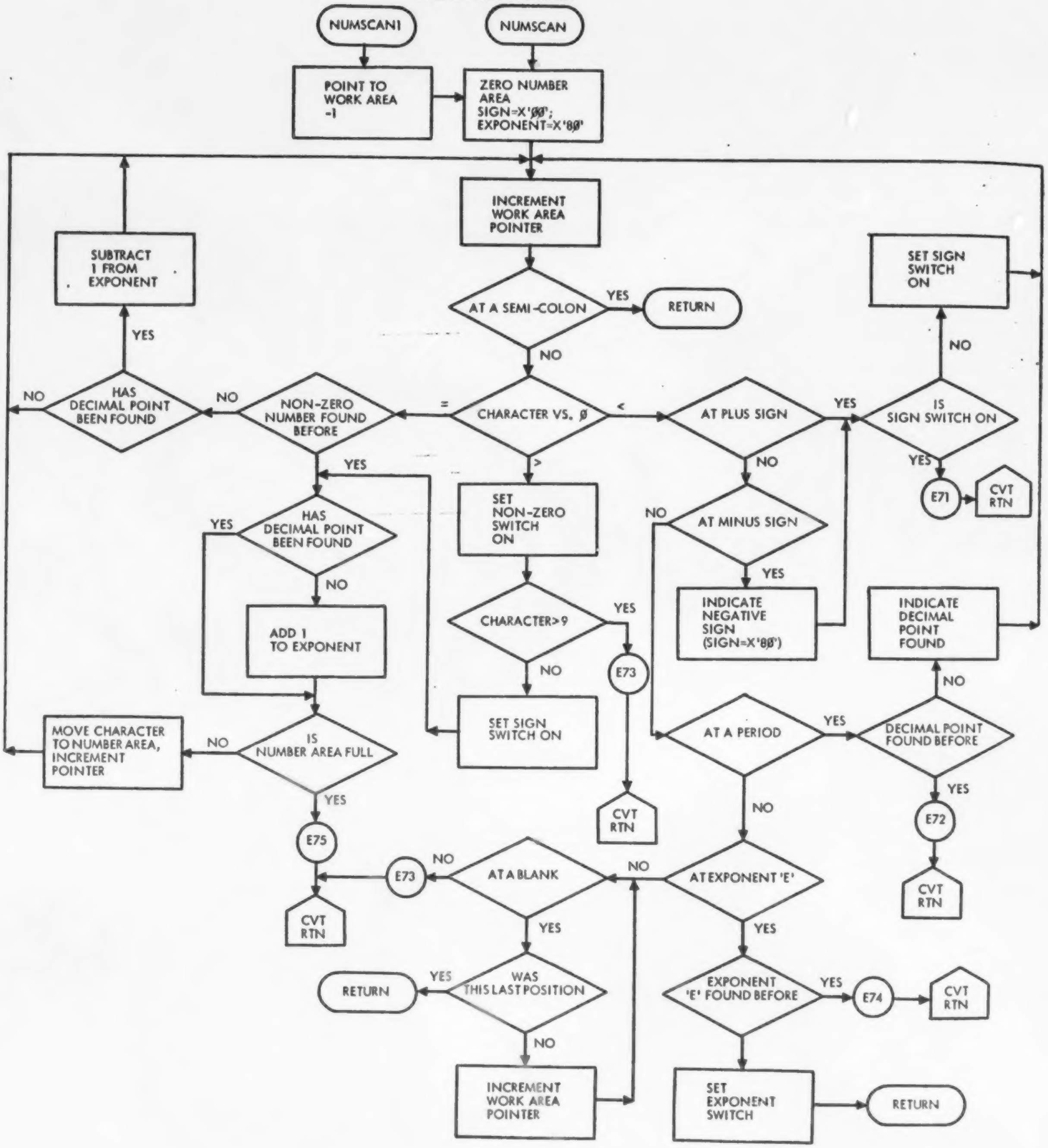

VALUECK
MOVE MANTISSA OF LOWER VALUE INTO M1
MOVE MANTISSA OF UPPER VALUE INTO M2
M1 SIGN=M2 SIGN
NO
M1 SIGN PLUS
YES
SET CONDITION CODE FOR HIGH
NO
SET CONDITION CODE FOR LOW
RETURN
YES
EXP1 VS. EXP2
EXP1<EXP2
EXP1>EXP2
EQUAL
M1 VS. M2
M1<M2
M1>M2
IS M1 SIGN PLUS
YES
SET CONDITION CODE FOR LOW
NO
SET CONDITION CODE FOR HIGH
EQUAL
SET CONDITION CODE FOR EQUAL
IS M1 SIGN PLUS
YES
SET CONDITION CODE FOR HIGH
NO
SET CONDITION CODE FOR LOW
RETURN

CONVERT- MANTISSA/EXPONENT CONVERSION ROUTINE

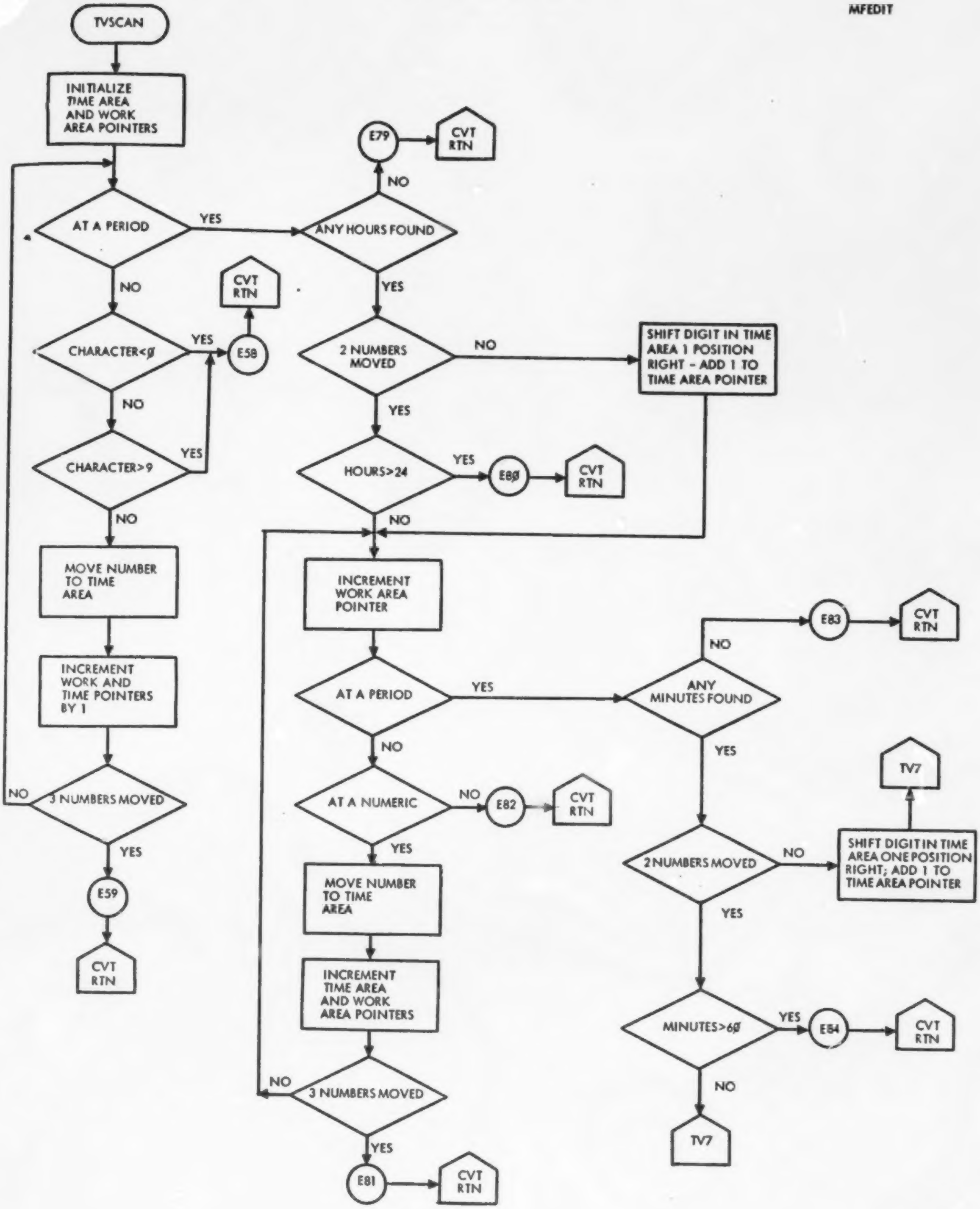
MFEDIT

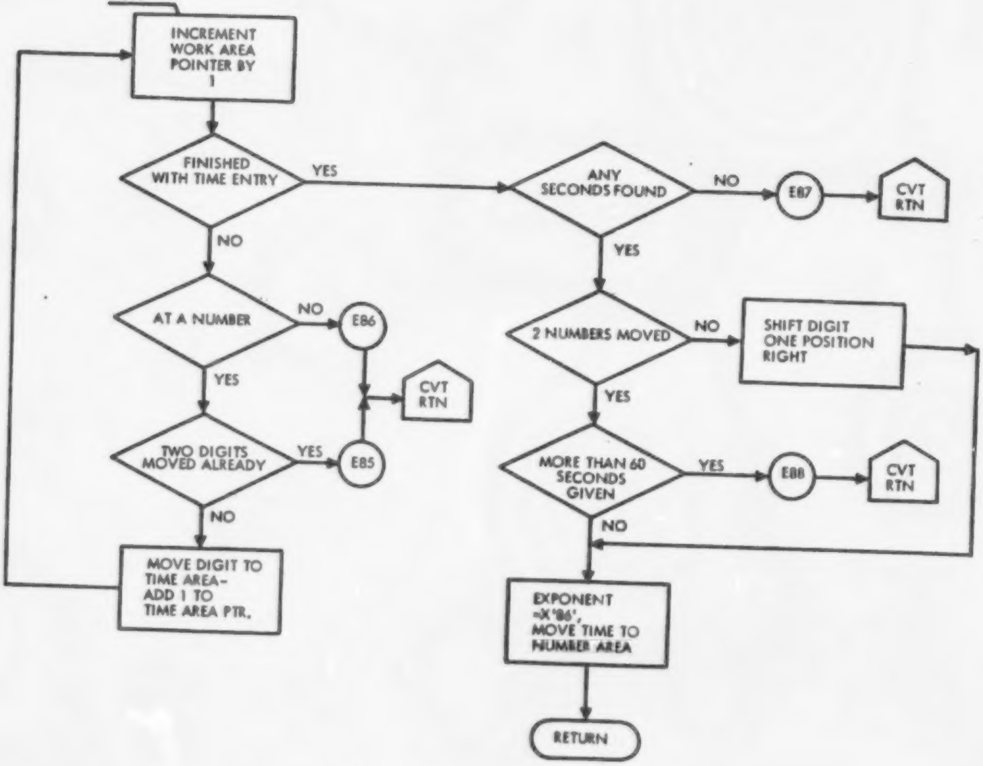

CNVRT
STORE REGISTERS 4-10; ZERO NUMBER AREA
IS DATE VALUE SWITCH ON
YES
NO
*DVSCAN
CHECK DATE VALUE
IS TIME VALUE SWITCH ON
YES
NO
TVSCAN
CHECK TIME VALUE
SIGN, PERIOD, E, AND NON-ZERO SWITCHES OFF
NUMSCAN1
CHECK MANTISSA PART OF NUMBER
IS SIGN SWITCH ON
NO
YES
E76
IS NUMBER AREA STILL 0
YES
NO
PACK NUMBER AND CONVERT TO BINARY
STORE THIS IN MANTISSA OF VALUE
IS E-SWITCH ON
NO
YES
RETURN
RESTORE REGISTERS 4 - 10
CVTRTN
INDICATE ERROR FOUND IN NUMBER
E78
YES
IS E-SWITCH ON
NO
SAVE X'80' AS EXPONENT
STORE EXPONENT IN SECOND BYTE OF VALUE
'OR' SIGN WITH TOP BYTE OF MANTISSA; MOVE VALUE CODE TO FIRST VALUE BYTE
0<EXP<255
YES
NO
ADD EXPONENT AND MANTISSA EXPONENT
COMPLEMENT THE EXPONENT
YES
IS SIGN OF EXPONENT MINUS
NO
NO
IS EXPONENT>255
YES
PACK EXPONENT AND CONVERT TO BINARY
NO
MORE THAN 3DIGITS FOUND
YES
YES
ANY EXPONENT FOUND
NO
E77
CVT RTN
NUMSCAN
CHECK EXPONENT PORTION OF NUM.
SIGN AND NON-ZERO SWITCHES OFF; PERIOD SW ON
SAVE MANTISSA SIGN AND EXPONENT

NUMSCAN- CHECKS FOR VALID MANTISSA AND EXPONENT ENTRIES

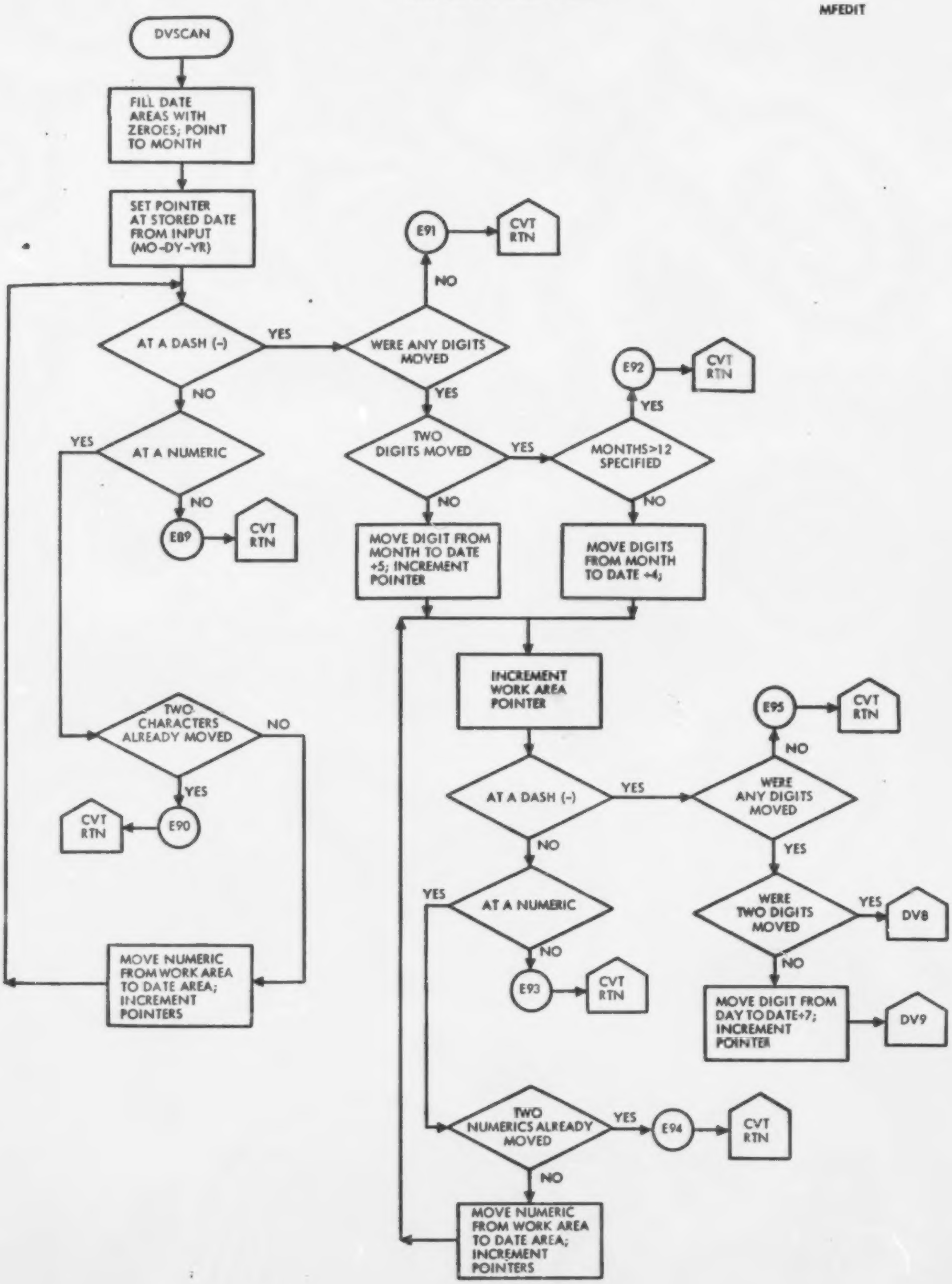

NUMSCAN1
NUMSCAN
POINT TO WORK AREA -1
ZERO NUMBER AREA SIGN=X'00'; EXPONENT=X'80'
INCREMENT WORK AREA POINTER
AT A SEMI-COLON
YES
RETURN
NO
CHARACTER VS. 0
=
<
>
NON-ZERO NUMBER FOUND BEFORE
NO
HAS DECIMAL POINT BEEN FOUND
NO
YES
SUBTRACT 1 FROM EXPONENT
YES
HAS DECIMAL POINT BEEN FOUND
YES
NO
ADD 1 TO EXPONENT
IS NUMBER AREA FULL
NO
MOVE CHARACTER TO NUMBER AREA, INCREMENT POINTER
YES
E75
CVT RTN
SET NON-ZERO SWITCH ON
CHARACTER>9
YES
E73
CVT RTN
NO
SET SIGN SWITCH ON
AT PLUS SIGN
YES
IS SIGN SWITCH ON
NO
SET SIGN SWITCH ON
YES
E71
CVT RTN
NO
AT MINUS SIGN
NO
YES
INDICATE NEGATIVE SIGN (SIGN=X'80')
AT A PERIOD
YES
DECIMAL POINT FOUND BEFORE
NO
INDICATE DECIMAL POINT FOUND
YES
E72
CVT RTN
NO
AT EXPONENT 'E'
NO
AT A BLANK
NO
E73
CVT RTN
YES
WAS THIS LAST POSITION
YES
RETURN
NO
INCREMENT WORK AREA POINTER
YES
EXPONENT 'E' FOUND BEFORE
YES
E74
CVT RTN
SET EXPONENT SWITCH
RETURN

TVSCAN- CHECK TIME VALUE

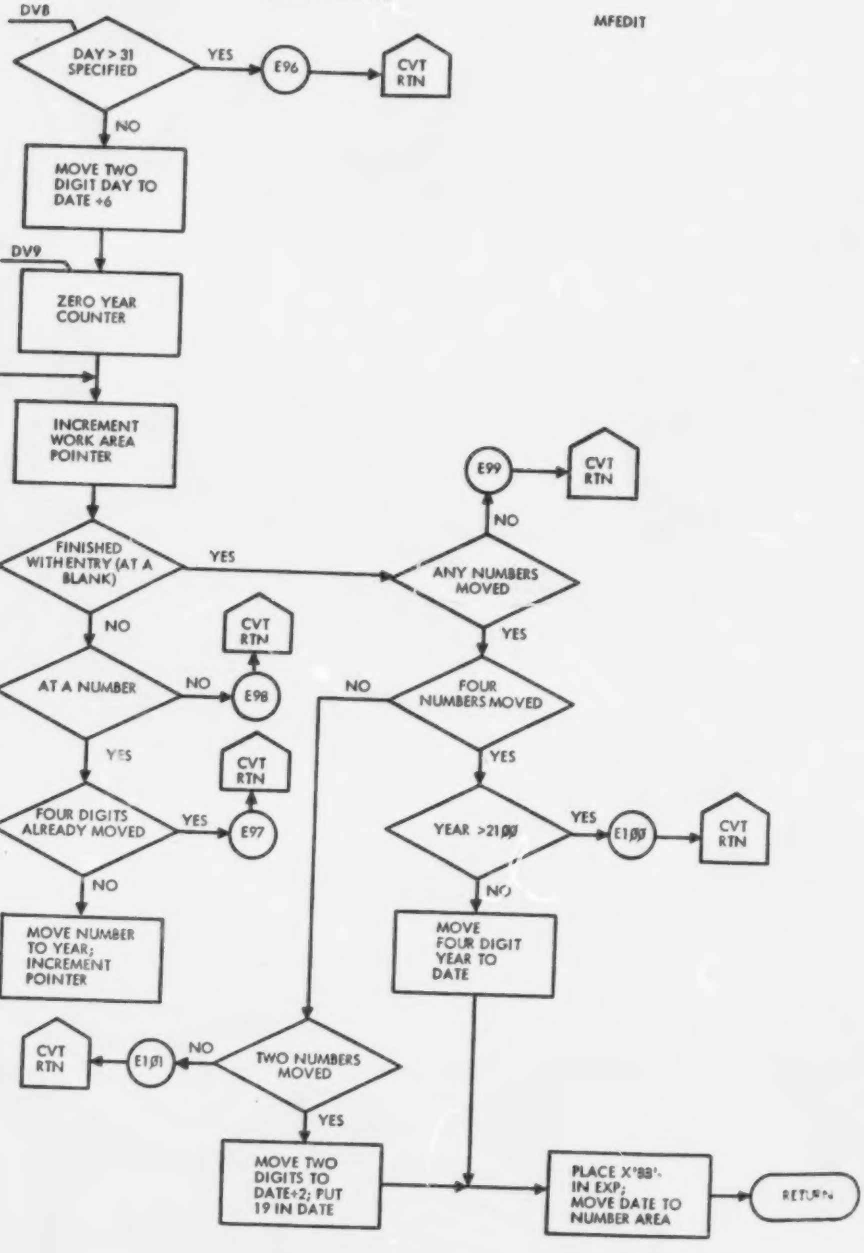

TVSCAN
INITIALIZE TIME AREA AND WORK AREA POINTERS
AT A PERIOD
YES
NO
CHARACTER<0
YES
E58
CVT RTN
NO
CHARACTER>9
YES
NO
MOVE NUMBER TO TIME AREA
INCREMENT WORK AND TIME POINTERS BY 1
3 NUMBERS MOVED
NO
YES
E59
CVT RTN
E79
CVT RTN
NO
ANY HOURS FOUND
YES
2 NUMBERS MOVED
NO
SHIFT DIGIT IN TIME AREA 1 POSITION RIGHT - ADD 1 TO TIME AREA POINTER
YES
HOURS>24
YES
E80
CVT RTN
NO
INCREMENT WORK AREA POINTER
AT A PERIOD
YES
NO
AT A NUMERIC
NO
E82
CVT RTN
YES
MOVE NUMBER TO TIME AREA
INCREMENT TIME AREA AND WORK AREA POINTERS
3 NUMBERS MOVED
NO
YES
E81
CVT RTN
E83
CVT RTN
NO
ANY MINUTES FOUND
YES
2 NUMBERS MOVED
NO
SHIFT DIGIT IN TIME AREA ONE POSITION RIGHT; ADD 1 TO TIME AREA POINTER
TV7
YES
MINUTES>60
YES
E84
CVT RTN
NO
TV7


TV7
INCREMENT WORK AREA POINTER BY 1
FINISHED WITH TIME ENTRY
YES
NO
AT A NUMBER
NO
E86
YES
CVT RTN
TWO DIGITS MOVED ALREADY
YES
E85
NO
MOVE DIGIT TO TIME AREA-ADD 1 TO TIME AREA PTR.
ANY SECONDS FOUND
NO
E87
CVT RTN
YES
2 NUMBERS MOVED
NO
SHIFT DIGIT ONE POSITION RIGHT
YES
MORE THAN 60 SECONDS GIVEN
YES
E88
CVT RTN
NO
EXPONENT =X'86', MOVE TIME TO NUMBER AREA
RETURN

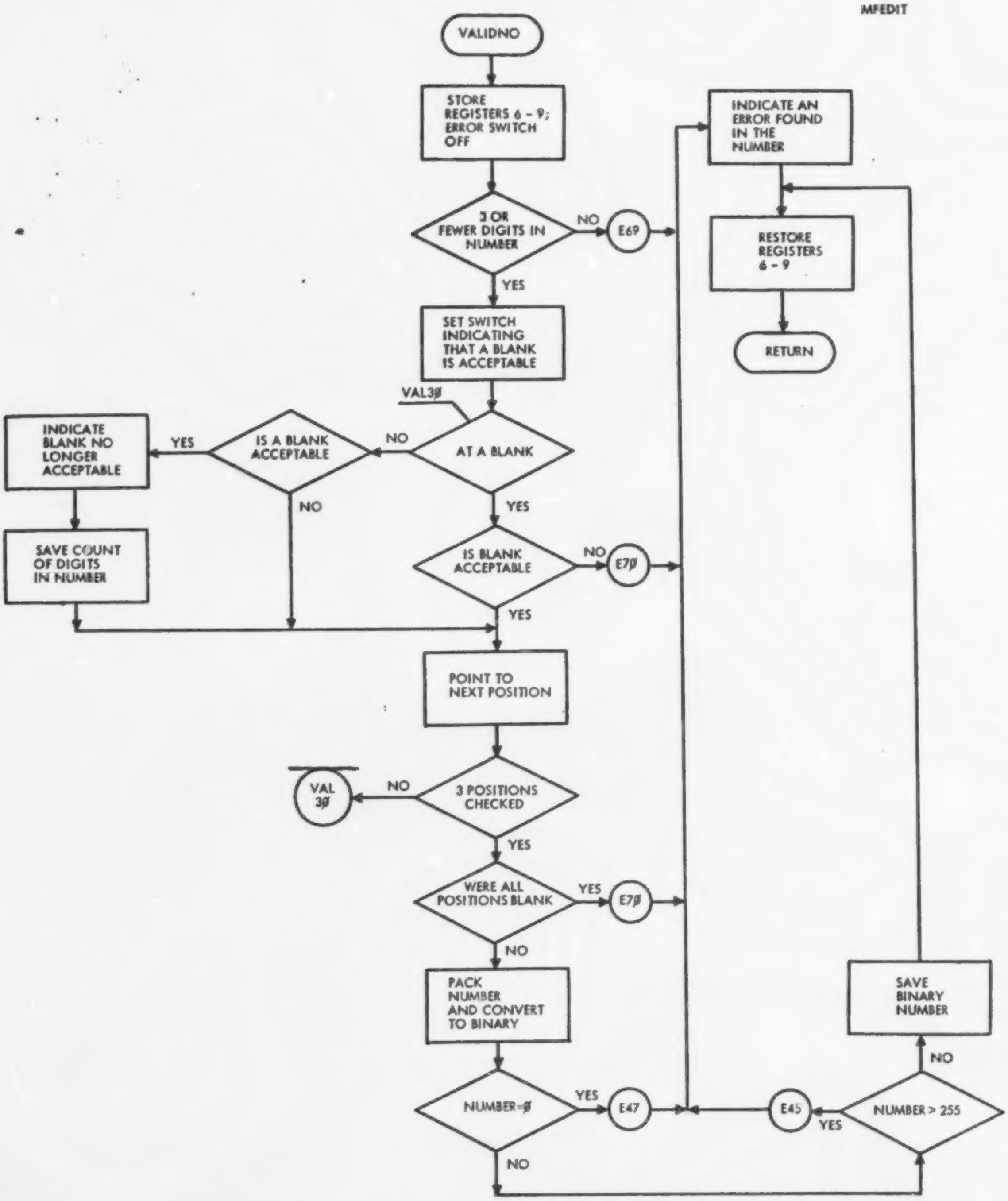
DVSCAN- CHECK DATE VALUES


DVSCAN
FILL DATE AREAS WITH ZEROES; POINT TO MONTH
SET POINTER AT STORED DATE FROM INPUT (MO-DY-YR)
AT A DASH (-)
YES
NO
AT A NUMERIC
YES
NO
E89
CVT RTN
TWO CHARACTERS ALREADY MOVED
NO
YES
E90
CVT RTN
MOVE NUMERIC FROM WORK AREA TO DATE AREA; INCREMENT POINTERS
WERE ANY DIGITS MOVED
NO
E91
CVT RTN
YES
TWO DIGITS MOVED
YES
NO
MONTHS>12 SPECIFIED
YES
E92
CVT RTN
NO
MOVE DIGIT FROM MONTH TO DATE +5; INCREMENT POINTER
MOVE DIGITS FROM MONTH TO DATE +4;
INCREMENT WORK AREA POINTER
AT A DASH (-)
YES
NO
WERE ANY DIGITS MOVED
NO
E95
CVT RTN
YES
WERE TWO DIGITS MOVED
YES
DV8
NO
MOVE DIGIT FROM DAY TO DATE+7; INCREMENT POINTER
DV9
AT A NUMERIC
YES
NO
E93
CVT RTN
TWO NUMERICS ALREADY MOVED
YES
E94
CVT RTN
NO
MOVE NUMERIC FROM WORK AREA TO DATE AREA; INCREMENT POINTERS

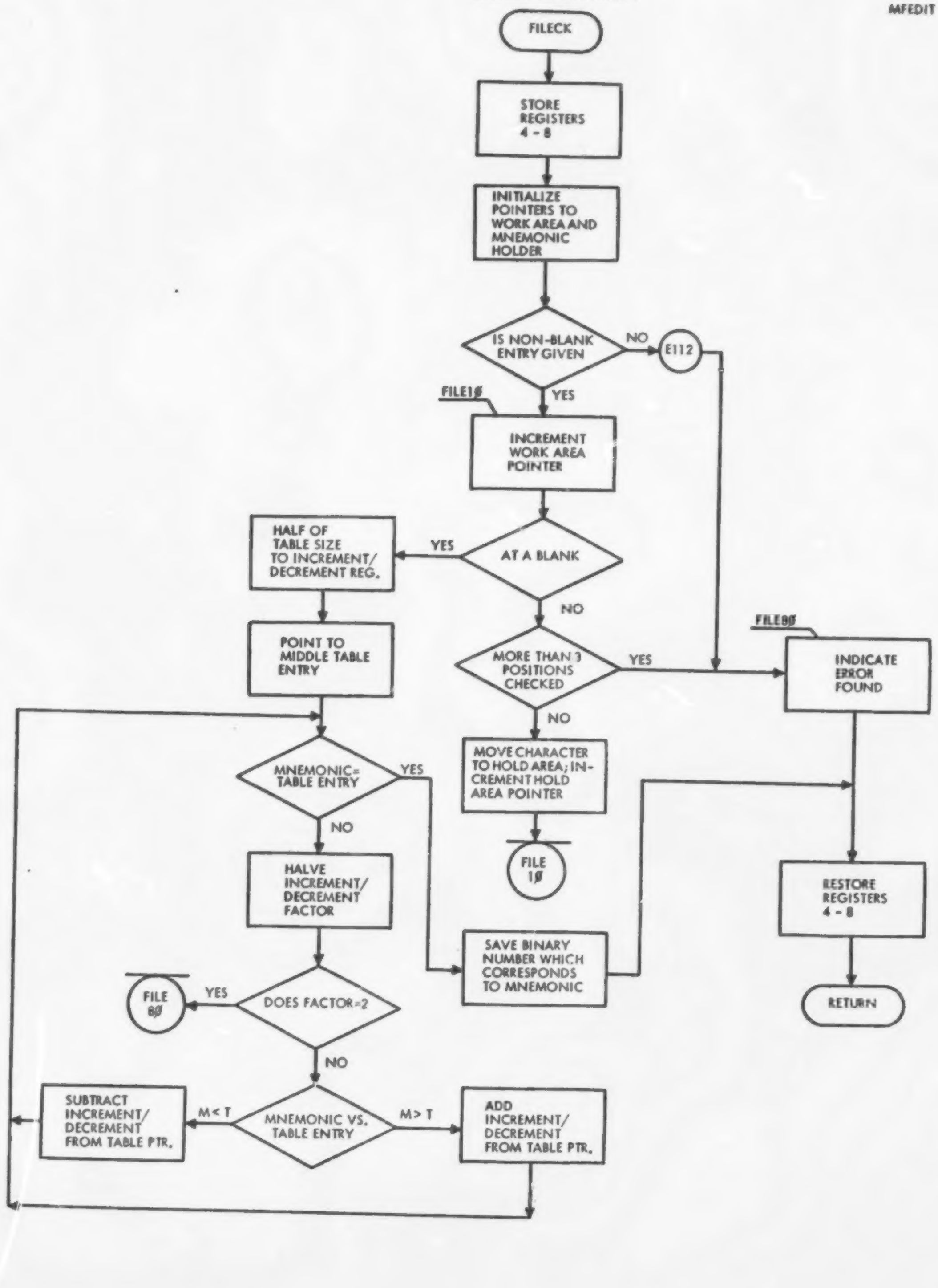
DVSCAN (CONT.)


DV8
DAY > 31 SPECIFIED
YES
E96
CVT RTN
NO
MOVE TWO DIGIT DAY TO DATE +6
DV9
ZERO YEAR COUNTER
INCREMENT WORK AREA POINTER
FINISHED WITH ENTRY (AT A BLANK)
YES
ANY NUMBERS MOVED
NO
E99
CVT RTN
YES
NO
AT A NUMBER
NO
E98
CVT RTN
YES
FOUR DIGITS ALREADY MOVED
YES
E97
CVT RTN
NO
MOVE NUMBER TO YEAR; INCREMENT POINTER
FOUR NUMBERS MOVED
NO
YES
YEAR >2100
YES
E100
CVT RTN
NO
MOVE FOUR DIGIT YEAR TO DATE
TWO NUMBERS MOVED
NO
E101
CVT RTN
YES
MOVE TWO DIGITS TO DATE+2; PUT 19 IN DATE
PLACE X'88' IN EXP; MOVE DATE TO NUMBER AREA
RETURN

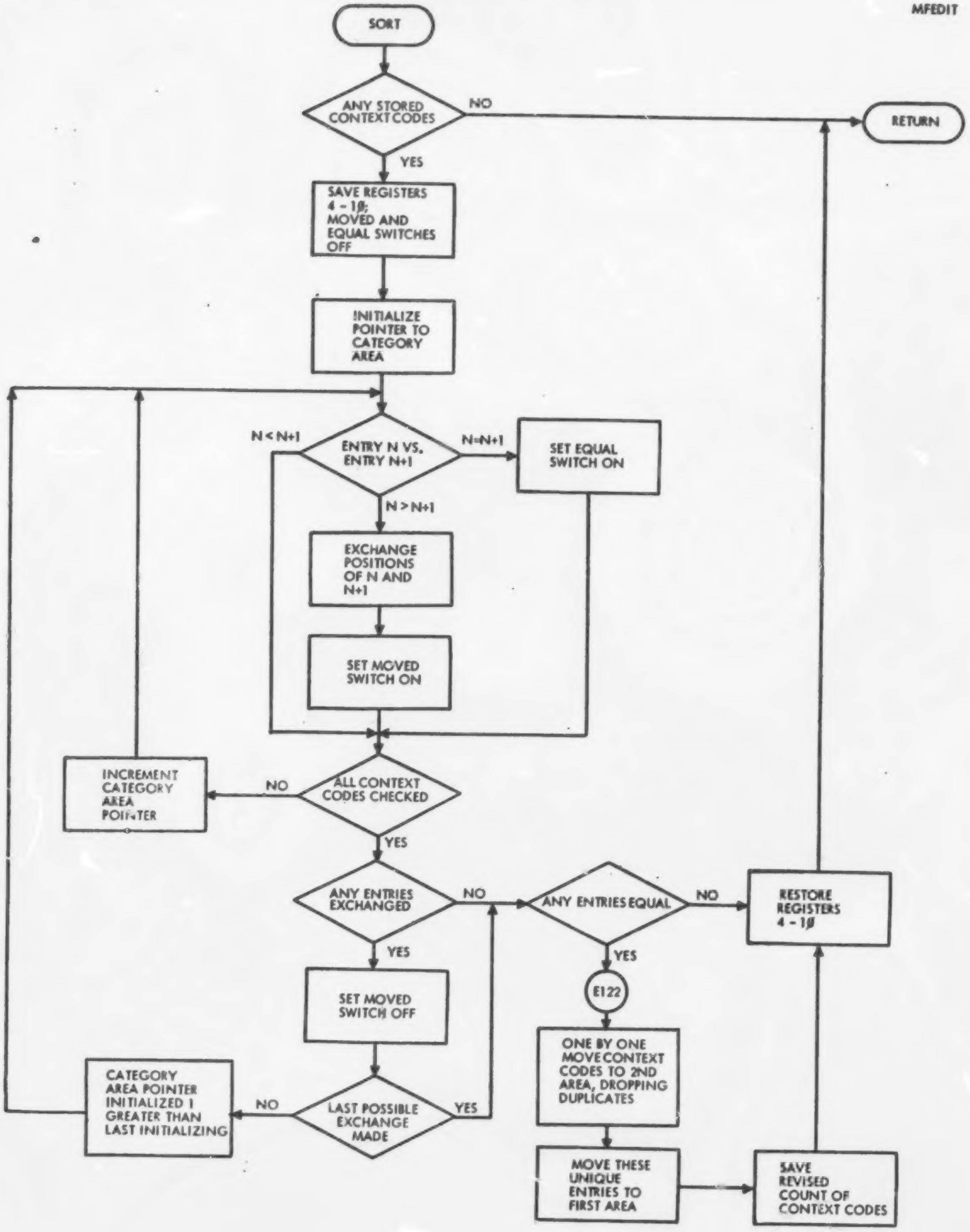
VALIDNO - CHECK DESCRIPTOR RANK AND TEXT SEGMENT NUMBERS


VALIDNO
STORE REGISTERS 6 - 9; ERROR SWITCH OFF
3 OR FEWER DIGITS IN NUMBER
NO
E69
YES
SET SWITCH INDICATING THAT A BLANK IS ACCEPTABLE
VAL30
AT A BLANK
NO
IS A BLANK ACCEPTABLE
YES
INDICATE BLANK NO LONGER ACCEPTABLE
NO
SAVE COUNT OF DIGITS IN NUMBER
YES
IS BLANK ACCEPTABLE
NO
E70
YES
POINT TO NEXT POSITION
3 POSITIONS CHECKED
NO
VAL 30
YES
WERE ALL POSITIONS BLANK
YES
E70
NO
PACK NUMBER AND CONVERT TO BINARY
NUMBER=0
YES
E47
NO
NUMBER > 255
YES
E45
NO
SAVE BINARY NUMBER
INDICATE AN ERROR FOUND IN THE NUMBER
RESTORE REGISTERS 6 - 9
RETURN

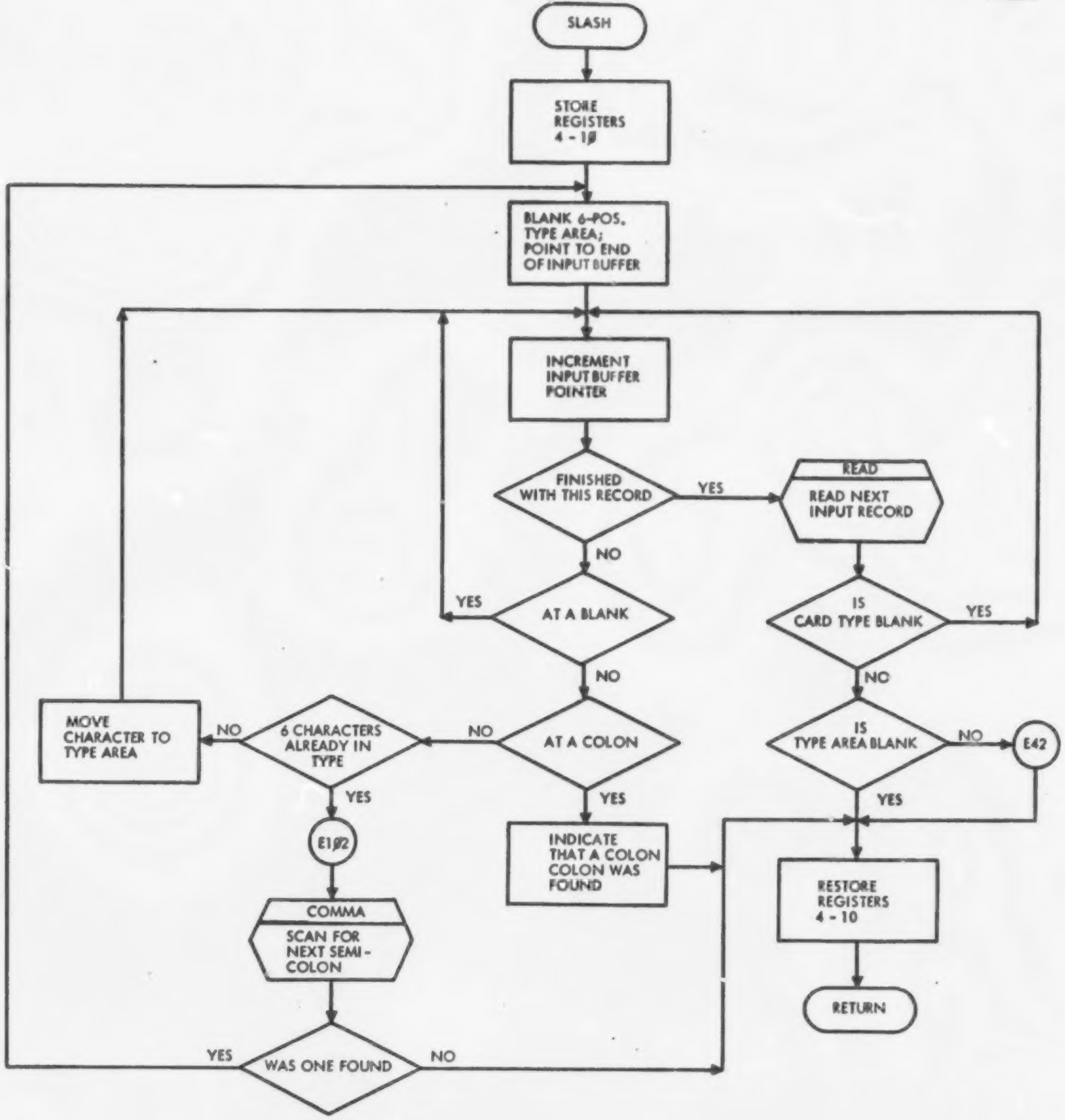
FILECK- CHECK FILE AND
CONTEXT CODE NUMBERS


FILECK
STORE REGISTERS 4 - 8
INITIALIZE POINTERS TO WORK AREA AND MNEMONIC HOLDER
IS NON-BLANK ENTRY GIVEN
NO
E112
YES
FILE10
INCREMENT WORK AREA POINTER
AT A BLANK
YES
HALF OF TABLE SIZE TO INCREMENT/ DECREMENT REG.
NO
POINT TO MIDDLE TABLE ENTRY
MORE THAN 3 POSITIONS CHECKED
YES
FILE80
INDICATE ERROR FOUND
NO
MOVE CHARACTER TO HOLD AREA; INCREMENT HOLD AREA POINTER
FILE 10
MNEMONIC= TABLE ENTRY
YES
NO
HALVE INCREMENT/ DECREMENT FACTOR
SAVE BINARY NUMBER WHICH CORRESPONDS TO MNEMONIC
RESTORE REGISTERS 4 - 8
RETURN
FILE 80
YES
DOES FACTOR=2
NO
SUBTRACT INCREMENT/ DECREMENT FROM TABLE PTR.
M< T
MNEMONIC VS. TABLE ENTRY
M> T
ADD INCREMENT/ DECREMENT FROM TABLE PTR.

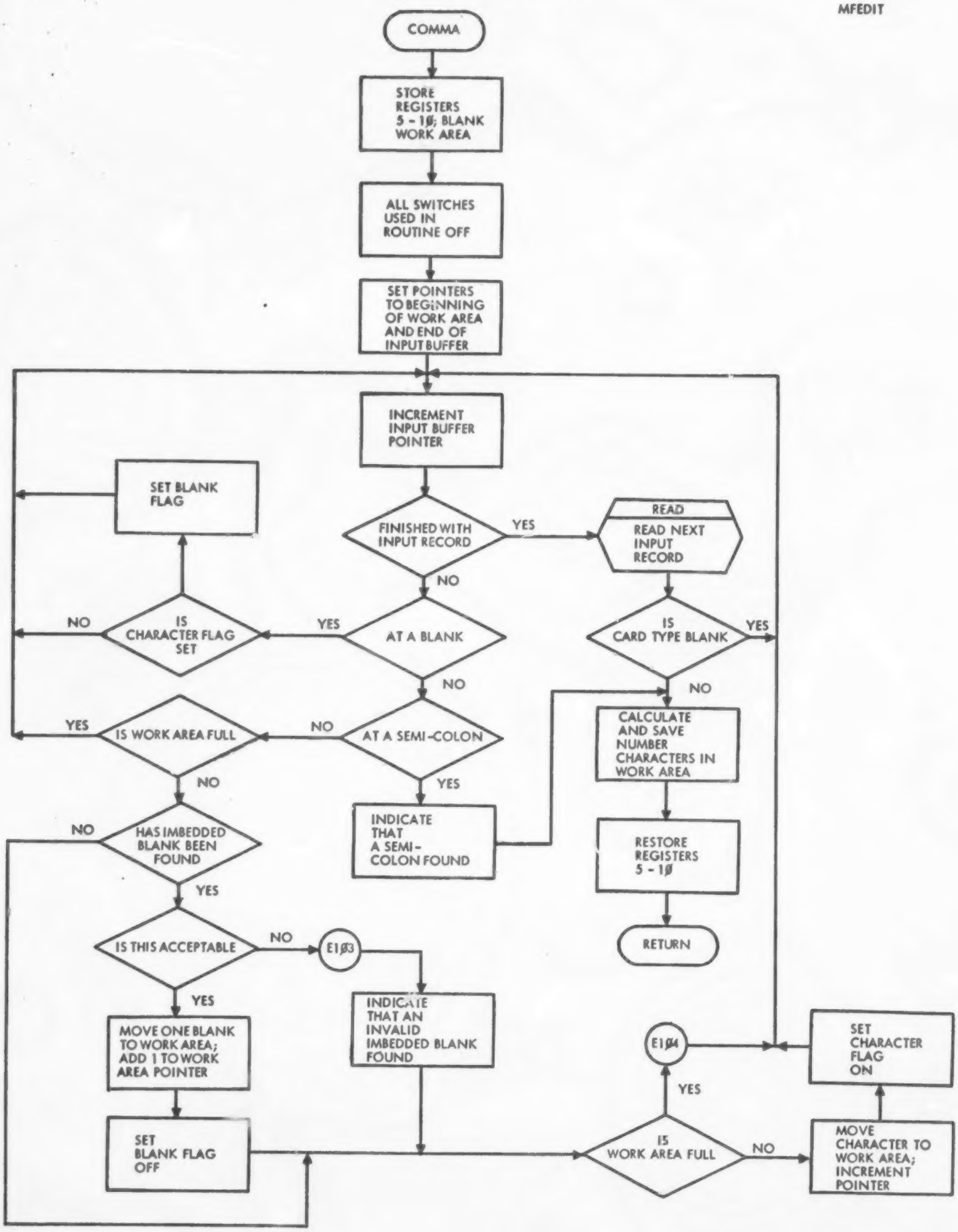
SORT- SORT BINARY CONTEXT CODES


SORT
ANY STORED CONTEXT CODES
NO
RETURN
YES
SAVE REGISTERS 4 - 10; MOVED AND EQUAL SWITCHES OFF
INITIALIZE POINTER TO CATEGORY AREA
N < N+1
ENTRY N VS. ENTRY N+1
N=N+1
SET EQUAL SWITCH ON
N > N+1
EXCHANGE POSITIONS OF N AND N+1
SET MOVED SWITCH ON
INCREMENT CATEGORY AREA POINTER
NO
ALL CONTEXT CODES CHECKED
YES
ANY ENTRIES EXCHANGED
NO
ANY ENTRIES EQUAL
NO
RESTORE REGISTERS 4 - 10
YES
SET MOVED SWITCH OFF
YES
E122
ONE BY ONE MOVE CONTEXT CODES TO 2ND AREA, DROPPING DUPLICATES
CATEGORY AREA POINTER INITIALIZED 1 GREATER THAN LAST INITIALIZING
NO
LAST POSSIBLE EXCHANGE MADE
YES
MOVE THESE UNIQUE ENTRIES TO FIRST AREA
SAVE REVISED COUNT OF CONTEXT CODES

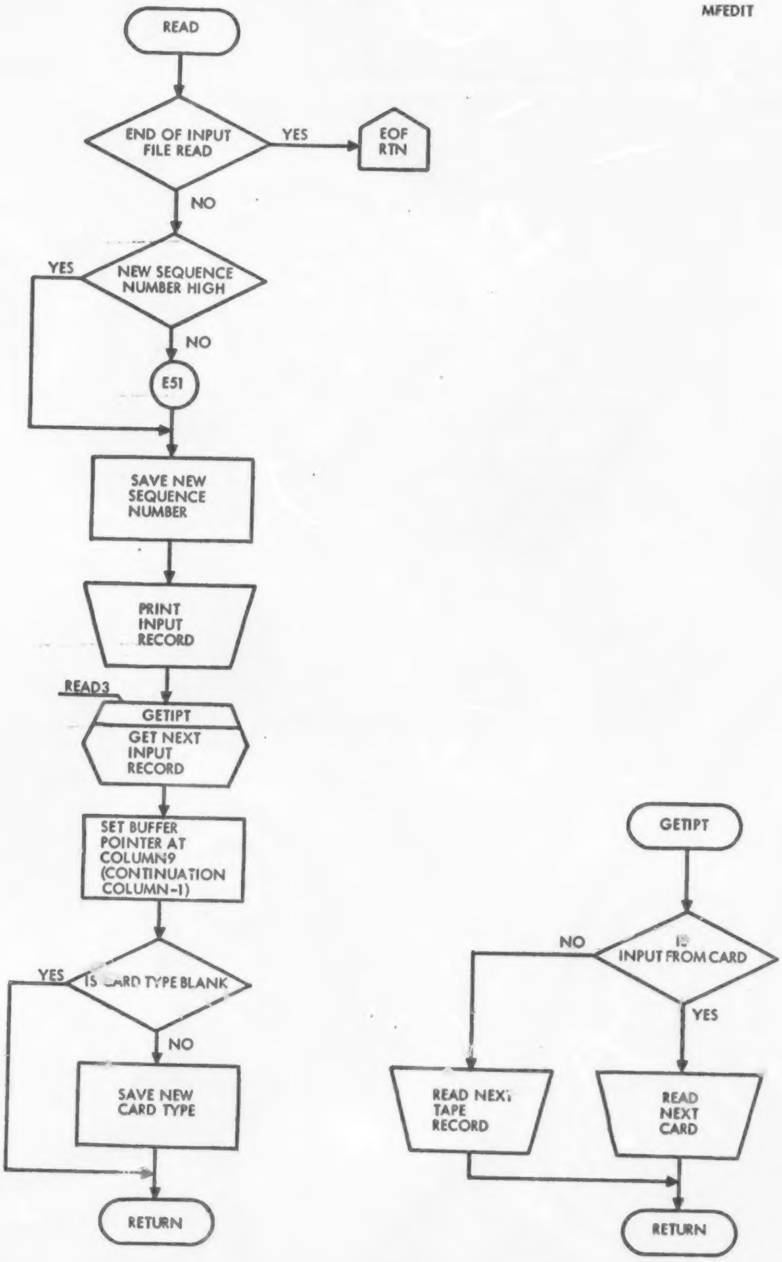
SLASH- SCAN FOR COLON


SLASH
STORE REGISTERS 4 - 10
BLANK 6-POS. TYPE AREA; POINT TO END OF INPUT BUFFER
INCREMENT INPUT BUFFER POINTER
FINISHED WITH THIS RECORD
YES
READ
READ NEXT INPUT RECORD
NO
YES
AT A BLANK
IS CARD TYPE BLANK
YES
NO
NO
MOVE CHARACTER TO TYPE AREA
NO
6 CHARACTERS ALREADY IN TYPE
NO
AT A COLON
IS TYPE AREA BLANK
NO
E42
YES
YES
YES
E102
INDICATE THAT A COLON COLON WAS FOUND
RESTORE REGISTERS 4 - 10
COMMA
SCAN FOR NEXT SEMI-COLON
RETURN
YES
WAS ONE FOUND
NO

COMMA- SCAN FOR SEMI-COLON

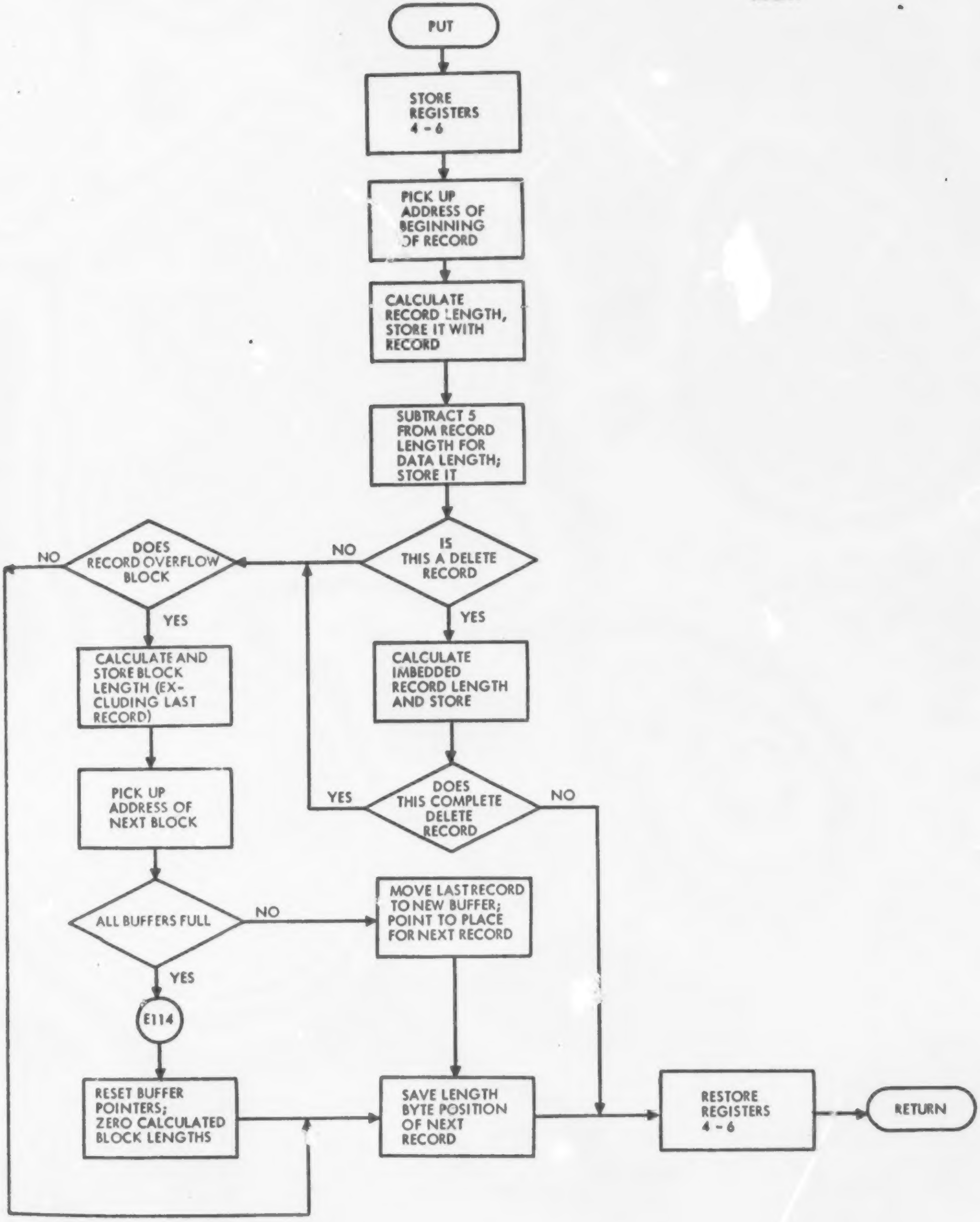

COMMA
STORE REGISTERS 5 - 10; BLANK WORK AREA
ALL SWITCHES USED IN ROUTINE OFF
SET POINTERS TO BEGINNING OF WORK AREA AND END OF INPUT BUFFER
INCREMENT INPUT BUFFER POINTER
FINISHED WITH INPUT RECORD
YES
READ
READ NEXT INPUT RECORD
NO
SET BLANK FLAG
IS CHARACTER FLAG SET
NO
YES
AT A BLANK
IS CARD TYPE BLANK
YES
NO
NO
YES
IS WORK AREA FULL
NO
AT A SEMI-COLON
CALCULATE AND SAVE NUMBER CHARACTERS IN WORK AREA
NO
YES
NO
HAS IMBEDDED BLANK BEEN FOUND
INDICATE THAT A SEMI-COLON FOUND
RESTORE REGISTERS 5 - 10
YES
IS THIS ACCEPTABLE
NO
E103
RETURN
YES
INDICATE THAT AN INVALID IMBEDDED BLANK FOUND
MOVE ONE BLANK TO WORK AREA; ADD 1 TO WORK AREA POINTER
E104
SET CHARACTER FLAG ON
YES
SET BLANK FLAG OFF
IS WORK AREA FULL
NO
MOVE CHARACTER TO WORK AREA; INCREMENT POINTER

READ- READ SEQUENCE CHECK
AND PRINT INPUT RECORDS


READ
END OF INPUT FILE READ
YES
EOF RTN
NO
YES
NEW SEQUENCE NUMBER HIGH
NO
E51
SAVE NEW SEQUENCE NUMBER
PRINT INPUT RECORD
READ3
GETIPT
GET NEXT INPUT RECORD
SET BUFFER POINTER AT COLUMN9 (CONTINUATION COLUMN-1)
YES
IS CARD TYPE BLANK
NO
SAVE NEW CARD TYPE
RETURN
GETIPT
NO
IS INPUT FROM CARD
YES
READ NEXT TAPE RECORD
READ NEXT CARD
RETURN

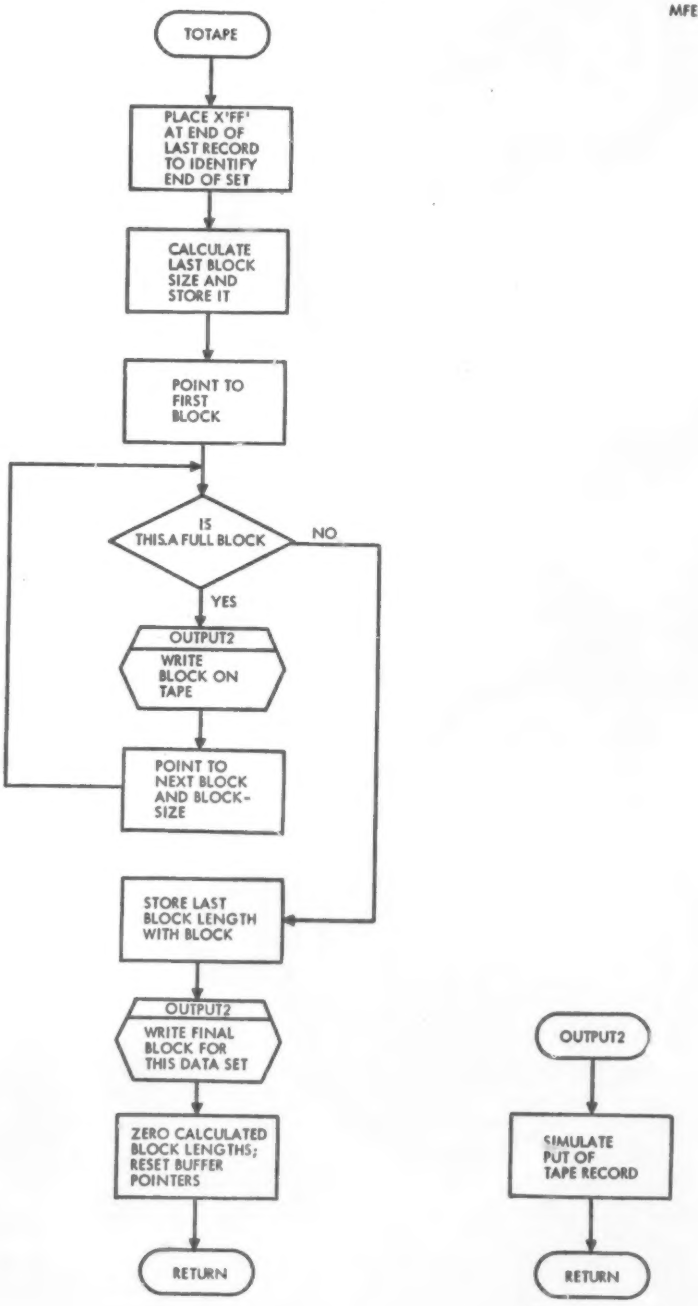
PUT- BUILD CORE RECORDS

MFEDIT

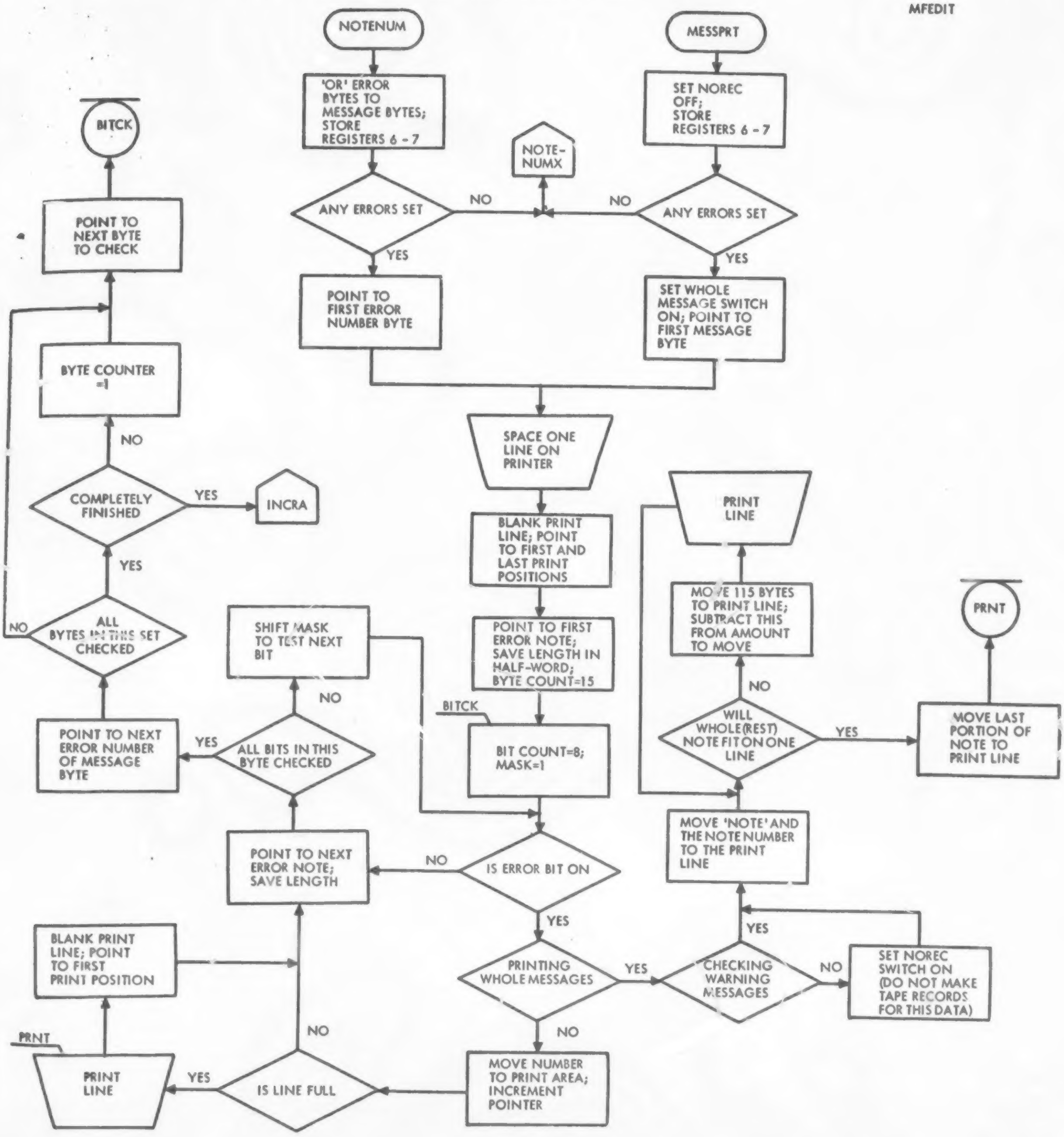

PUT
STORE REGISTERS 4 - 6
PICK UP ADDRESS OF BEGINNING OF RECORD
CALCULATE RECORD LENGTH, STORE IT WITH RECORD
SUBTRACT 5 FROM RECORD LENGTH FOR DATA LENGTH; STORE IT
IS THIS A DELETE RECORD
NO
YES
DOES RECORD OVERFLOW BLOCK
NO
YES
CALCULATE IMBEDDED RECORD LENGTH AND STORE
CALCULATE AND STORE BLOCK LENGTH (EX-CLUDING LAST RECORD)
DOES THIS COMPLETE DELETE RECORD
YES
NO
PICK UP ADDRESS OF NEXT BLOCK
ALL BUFFERS FULL
NO
YES
MOVE LAST RECORD TO NEW BUFFER; POINT TO PLACE FOR NEXT RECORD
E114
RESET BUFFER POINTERS; ZERO CALCULATED BLOCK LENGTHS
SAVE LENGTH BYTE POSITION OF NEXT RECORD
RESTORE REGISTERS 4 - 6
RETURN

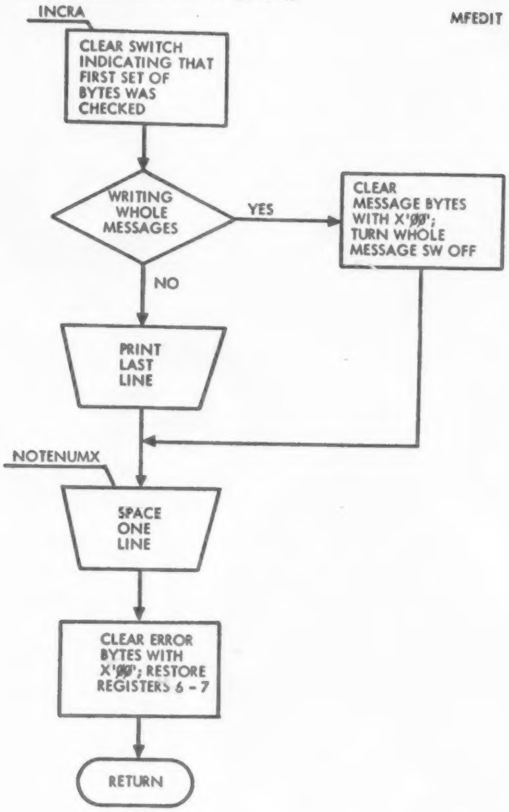
TOTAPE- WRITE CORE BUFFERS ON TAPE

MFEDIT


TOTAPE
PLACE X'FF' AT END OF LAST RECORD TO IDENTIFY END OF SET
CALCULATE LAST BLOCK SIZE AND STORE IT
POINT TO FIRST BLOCK
IS THIS A FULL BLOCK
NO
YES
OUTPUT2
WRITE BLOCK ON TAPE
POINT TO NEXT BLOCK AND BLOCK-SIZE
STORE LAST BLOCK LENGTH WITH BLOCK
OUTPUT2
WRITE FINAL BLOCK FOR THIS DATA SET
ZERO CALCULATED BLOCK LENGTHS; RESET BUFFER POINTERS
RETURN
OUTPUT2
SIMULATE PUT OF TAPE RECORD
RETURN

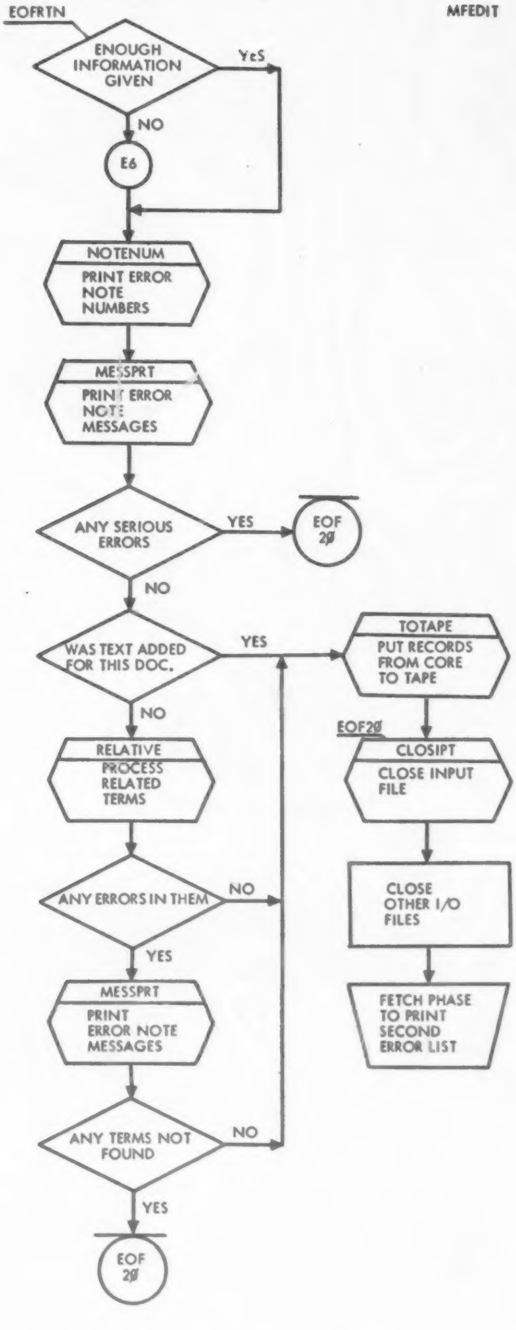
NOTENUM AND MESSPRT-PRINT ERROR NOTE NUMBERS AND MESSAGES

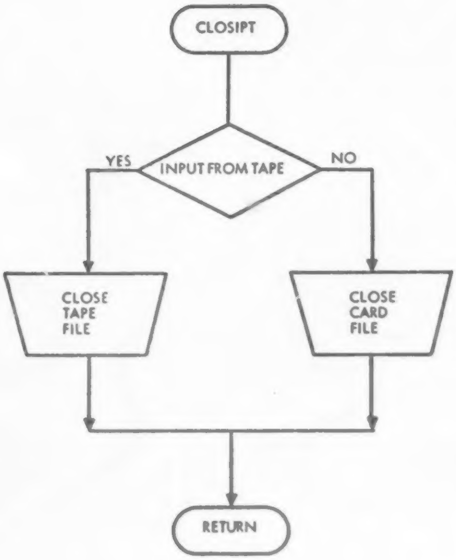

NOTENUM
'OR' ERROR BYTES TO MESSAGE BYTES; STORE REGISTERS 6 - 7
ANY ERRORS SET
NO
YES
POINT TO FIRST ERROR NUMBER BYTE
NOTE-NUMX
MESSPRT
SET NOREC OFF; STORE REGISTERS 6 - 7
ANY ERRORS SET
NO
YES
SET WHOLE MESSAGE SWITCH ON; POINT TO FIRST MESSAGE BYTE
SPACE ONE LINE ON PRINTER
BLANK PRINT LINE; POINT TO FIRST AND LAST PRINT POSITIONS
POINT TO FIRST ERROR NOTE; SAVE LENGTH IN HALF-WORD; BYTE COUNT=15
BITCK
BIT COUNT=8; MASK=1
IS ERROR BIT ON
NO
YES
POINT TO NEXT ERROR NOTE; SAVE LENGTH
ALL BITS IN THIS BYTE CHECKED
NO
YES
SHIFT MASK TO TEST NEXT BIT
POINT TO NEXT ERROR NUMBER OF MESSAGE BYTE
ALL BYTES IN THIS SET CHECKED
NO
YES
COMPLETELY FINISHED
YES
INCRA
NO
BYTE COUNTER =1
POINT TO NEXT BYTE TO CHECK
BITCK
PRINTING WHOLE MESSAGES
YES
NO
MOVE NUMBER TO PRINT AREA; INCREMENT POINTER
IS LINE FULL
YES
NO
PRNT
PRINT LINE
BLANK PRINT LINE; POINT TO FIRST PRINT POSITION
CHECKING WARNING MESSAGES
YES
NO
SET NOREC SWITCH ON (DO NOT MAKE TAPE RECORDS FOR THIS DATA)
MOVE 'NOTE' AND THE NOTE NUMBER TO THE PRINT LINE
WILL WHOLE(REST) NOTE FIT ON ONE LINE
NO
YES
MOVE 115 BYTES TO PRINT LINE; SUBTRACT THIS FROM AMOUNT TO MOVE
PRINT LINE
MOVE LAST PORTION OF NOTE TO PRINT LINE
PRNT


INCRA
CLEAR SWITCH INDICATING THAT FIRST SET OF BYTES WAS CHECKED
WRITING WHOLE MESSAGES
YES
CLEAR MESSAGE BYTES WITH X'00'; TURN WHOLE MESSAGE SW OFF
NO
PRINT LAST LINE
NOTENUMX
SPACE ONE LINE
CLEAR ERROR BYTES WITH X'00'; RESTORE REGISTERS 6 - 7
RETURN

EOFRTN- DO LAST PROCESSING

MFEDIT


EOFRTN
ENOUGH INFORMATION GIVEN
YES
NO
E6
NOTENUM
PRINT ERROR NOTE NUMBERS
MESSPRT
PRINT ERROR NOTE MESSAGES
ANY SERIOUS ERRORS
YES
EOF 20
NO
WAS TEXT ADDED FOR THIS DOC.
YES
TOTAPE
PUT RECORDS FROM CORE TO TAPE
NO
EOF20
CLOSIPT
CLOSE INPUT FILE
RELATIVE
PROCESS RELATED TERMS
ANY ERRORS IN THEM
NO
CLOSE OTHER I/O FILES
YES
MESSPRT
PRINT ERROR NOTE MESSAGES
FETCH PHASE TO PRINT SECOND ERROR LIST
ANY TERMS NOT FOUND
NO
YES
EOF 20


CLOSIPT
YES
INPUT FROM TAPE
NO
CLOSE TAPE FILE
CLOSE CARD FILE
RETURN

MASTER FILE MAINTENANCE PROGRAM (MFMAINT)PROGRAM LOGIC

MFMAINT gets one logical input record at a time and applies it to the document specified by a HDR (Header) record. All changes pertaining to a given document follow its HDR record. There are three types of HDR records.

a.	HDR ADD	(add a new document)	(external number)
b.	HDR DEL	(delete an existing document)	(internal number)
c.	HDR REV	(revise an existing document)	(internal number)

Any combination of changes (additions, deletions, revisions) can follow a HDR record. MFMAINT processes the HDR records as follows:

a. HDR ADD (external No.)

Looks up the external number in the Cross-Reference File. Generates an error record if it is found, unless the x-ref override option is specified. If not in error the program assigns the next sequential internal number to this document. Creates a record to add this document to the Cross-Reference File for the Cross-Reference File update program. Writes this document number in the Cross-Reference File for subsequent look-ups. Builds the fixed portion of the master record for this document. Applies the logical records which follow it to this document number until another HDR record is encountered.

b. HDR DEL (internal No.)

Looks up the document in the Master File. Generates an error record if not found. If found, creates a record to delete it from the Cross-Reference File. Deletes it from the Master File and Cross-Reference File. Creates records to delete all descriptors in this document from the Inverted file. Creates a record to delete this document from the ID file.

c. HDR REV (internal No.)

Looks up this document in the Master file. Generates an error record if not found. Applies the logical records which follow it to this document until the next HDR record is encountered.

The records following any given HDR records all pertain to the same document and are processed as follows:

1.	Add Descriptor	The descriptors in the master record are compared with the input descriptor one at a time. If a match is found, the input record is ignored. If not, the descriptor is added to the record. The program then examines the next input record.

2. Add Categories	Input categories are compared with categories in the master record. The ones that are equal are ignored, and the ones that are not equal are merged in with the categories in the master record. The next input record is then examined.
3. Add Links	If a descriptor was added to the record, two bytes of input links are moved into the record, following the descriptor. If the descriptor was already present in the record and the links were present, two bytes of input links are logically OR'ed into the record. If the descriptor was already present but the links were not present, two bytes of input links are moved into the record. The next record is then examined.
4. Add Values	If a descriptor was added to the record, input values are moved into the record, following the descriptor. Otherwise, the input values that do not match with the values in the record are added at the end of the values present, if any, in the master record. The next input record is then examined.
5. Add Subdescriptor	If a descriptor was added to the record, the input descriptor is merely moved into the record. Otherwise, the subdescriptors present, if any, are compared with the input subdescriptor. If a match is found, the input descriptor is ignored, otherwise the input subdescriptor is moved into the record. The next input record is then examined. The above procedures are repeated if it is another add record. If it is a delete record it is processed as follows:

Delete Records

1. Delete Descriptor	The input descriptor is compared with the descriptors in the master record. If a match is found, the descriptor plus all its qualifiers are deleted from the record. If a match is not found, an error record is generated.
2. Delete Subdescriptor	The descriptor that the input subdescriptor is associated with is looked up in the master record. If a match is not found, an error record is generated. If a match is found, the input subdescriptor is looked up in the record.

	If a match is not found, an error record is generated; otherwise, the subdescriptor plus all values associated with it are deleted.
3. Delete Links	The descriptor that the input links are associated with is looked up in the record. If a match is not found, an error record is generated. Otherwise, the input record is checked for the presence of links. If present, the links are deleted; otherwise, an error record is generated.
4. Delete Categories	The descriptor with which the categories are associated is looked up in the record. If a match is not found, an error record is generated. Otherwise, input categories are looked up one at a time in the input record. If any one of them has no match, an error record is generated and input categories are ignored. Otherwise, the specified ones are deleted from the record.
5. Delete Values	The descriptor and/or subdescriptor that the input values are associated with is looked up in the record. If no match is found, an error record is generated. Otherwise, the input values are looked up one at a time in the record. If any one of them is not present in the record, an error record is generated and all the input values are ignored. Otherwise the specified ones are deleted from the record.

Revise Records

1. Revise Title	Title information from the input record is moved into the title field of the master record.

After all the input records pertaining to the searchable section of a document have been processed, it is stored in the next available address in the master file. The next record is then examined.

Add and/or Delete Text

When a document is looked up in the master file, the program reads all the text segments following the searchable section and stores them on a disk. In the process a table is built of all segment numbers and their disk addresses. When adding a text segment, this table is searched for the segment number being added. If found the segment number is ignored, otherwise, the segment number is added to the table in sort and this segment is written on the next available track.

When deleting a text segment, the table is searched for the segment number. An error record is generated if not found; otherwise, the table entry is tagged as deleted.

After all the processing is complete for one document, text segments that are not deleted are read from the disk and written in the master file following the searchable section at the end-of-file.

When an entire document is deleted or a new document is added to the file, a delete or an add record is created for the cross reference update program.

The program creates records to update other files of the system as follows: Whenever a descriptor is added to (deleted from) a document, a record is generated to add (delete) that document in the inverted file for that descriptor. Those records are written on tape which becomes the input tape to the inverted file update program.

After all the input records pertaining to the searchable section of a document have been processed and the document stored, a test is made to determine if any ID. file descriptor was added, deleted, or changed in that document. If so, all the non ID. file descriptors are deleted from the document and the document is written on tape. This tape is later used as input to the ID. file update program.

All documents that are added, deleted, or changed are written on a tape. The error records are also written on this tape, which becomes input to the master file print program.

MFMAINT processes input records in the manner described above until an end-of-file is encountered on the input tape. At this point, the directories and control information necessary for the next update run (i.e., highest internal number used during this run, the next available address, etc) is written on the data cell. The program is then terminated.

Input

The tape created by MFEDIT (format M1 below) serves as input to MFMAINT.

Output

Four tapes and a data cell are output by MFMAINT.

1. Tapes

   a. The tape which, after sorting, will provide input to the inverted file update program is written in the format shown below labelled M3. The records are fixed length and blocked.

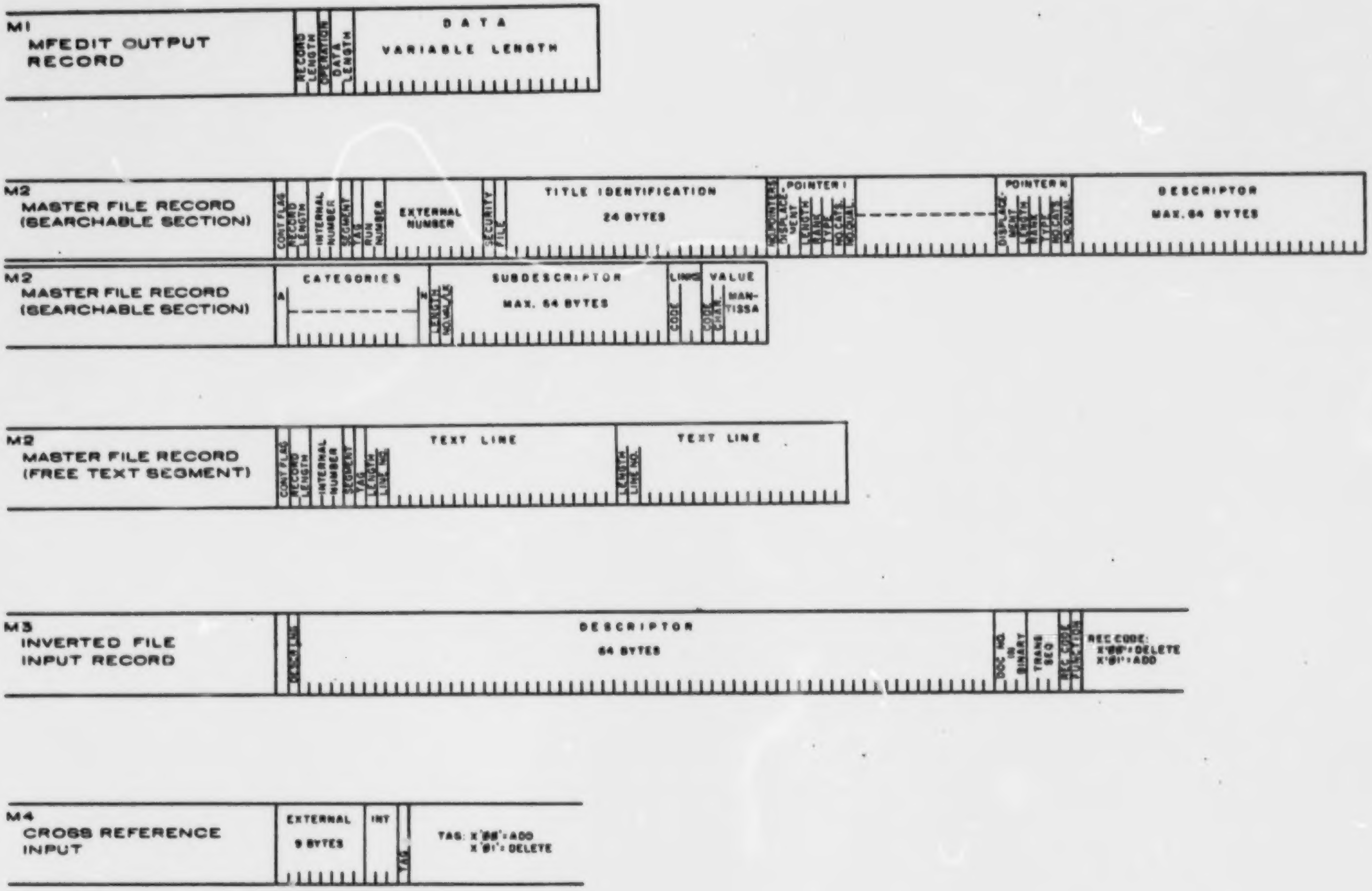
   b. The tape which, after sorting, will become input to the cross reference update program is written in the format shown below labelled M4. The records are fixed length and blocked.

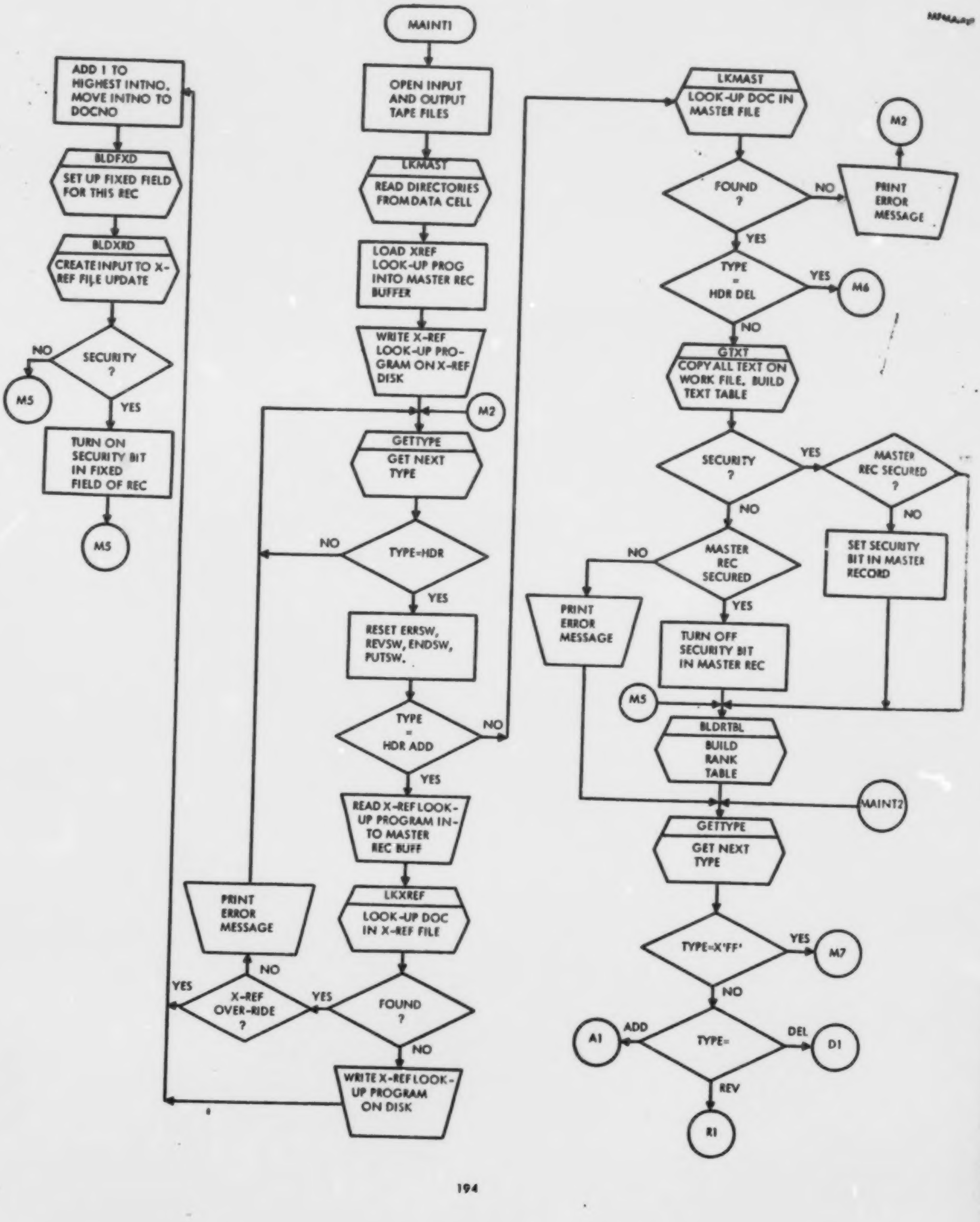
   c. Update to the ID. file is written in the format labelled M2 below. No free text is included in the ID. file.

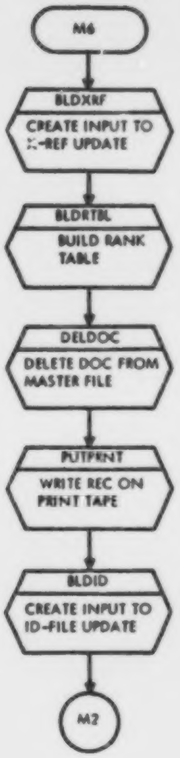
2. Data Cell

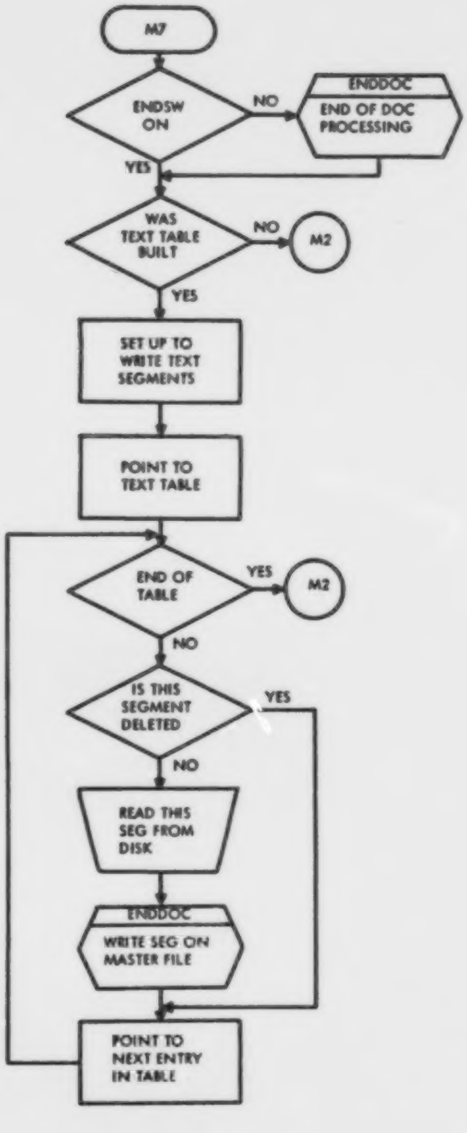
   The master file is written on the data cell in the format labelled M2 below. For a more detailed description of the master file record, please refer to Appendix B.

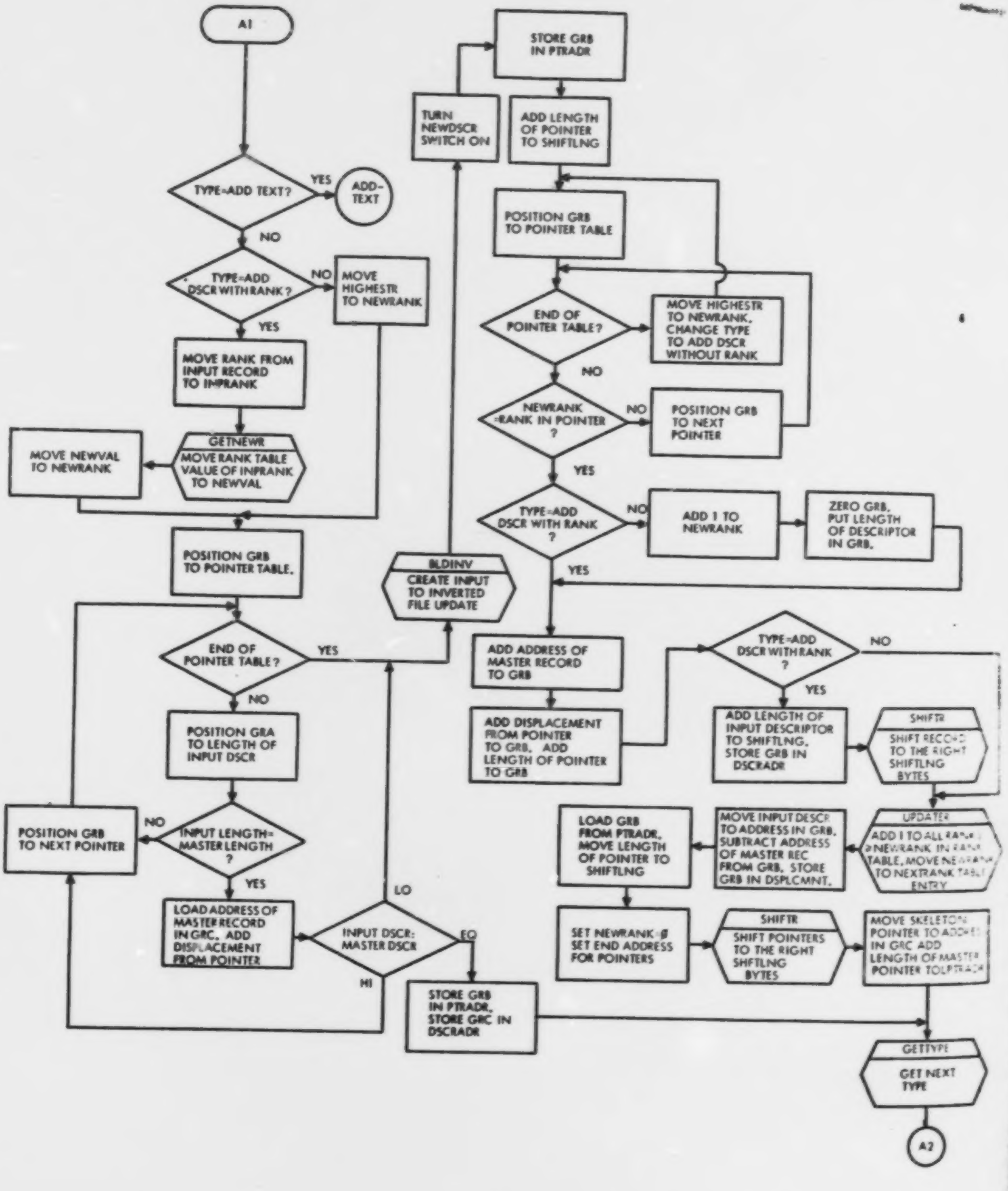
MASTER FILE MAINTENANCE PROGRAM (MFMAINT) RECORD FORMATS

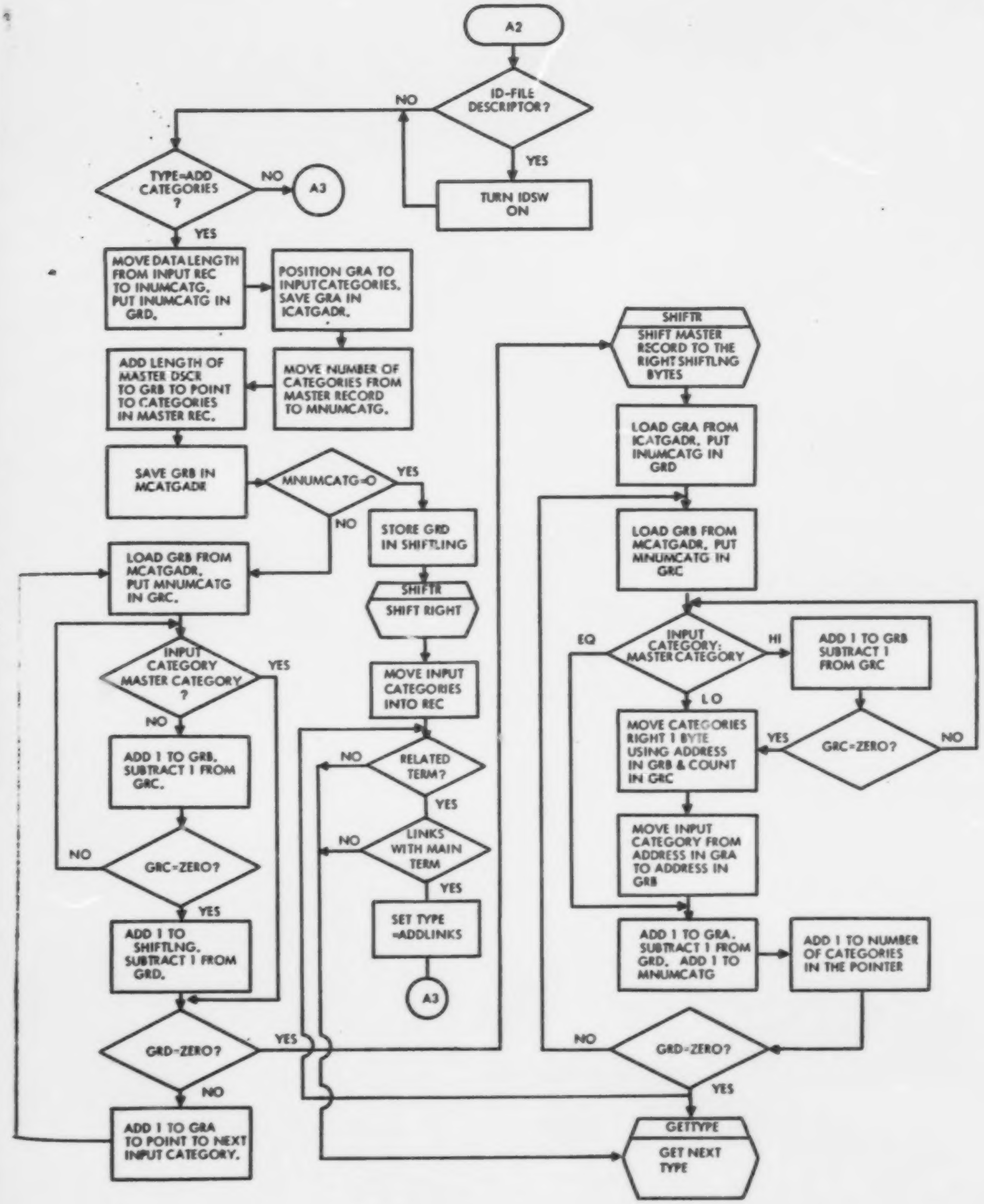

M1
MFEDIT OUTPUT RECORD
RECORD LENGTH
OPERATION
DATA LENGTH
DATA
VARIABLE LENGTH
M2
MASTER FILE RECORD (SEARCHABLE SECTION)
CONT FLAG
RECORD LENGTH
INTERNAL NUMBER
SEGMENT
TAG
RUN NUMBER
EXTERNAL NUMBER
SECURITY
FILE
TITLE IDENTIFICATION
24 BYTES
NO. POINTERS
POINTER I
DISPLACE-MENT
LENGTH
RANK
TYPE
NO. CATS.
NO. QUAL.
POINTER N
DISPLACE-MENT
LENGTH
RANK
TYPE
NO. CATS.
NO. QUAL.
DESCRIPTOR
MAX. 64 BYTES
M2
MASTER FILE RECORD (SEARCHABLE SECTION)
CATEGORIES
A
N
LENGTH
NO. VALUE
SUBDESCRIPTOR
MAX. 64 BYTES
LINK
CODE
VALUE
CODE
CHAR.
MAN-TISSA
M2
MASTER FILE RECORD (FREE TEXT SEGMENT)
CONT FLAG
RECORD LENGTH
INTERNAL NUMBER
SEGMENT
TAG
LENGTH
LINE NO.
TEXT LINE
LENGTH
LINE NO.
TEXT LINE
M3
INVERTED FILE INPUT RECORD
DESCRIPTOR
DESCRIPTOR
64 BYTES
DOC NO. IN BINARY
TRANS SEQ
REC CODE
FUNCTION
REC CODE:
X'80': DELETE
X'81': ADD
M4
CROSS REFERENCE INPUT
EXTERNAL
9 BYTES
INT
TAG
TAG: X'80': ADD
X'81': DELETE

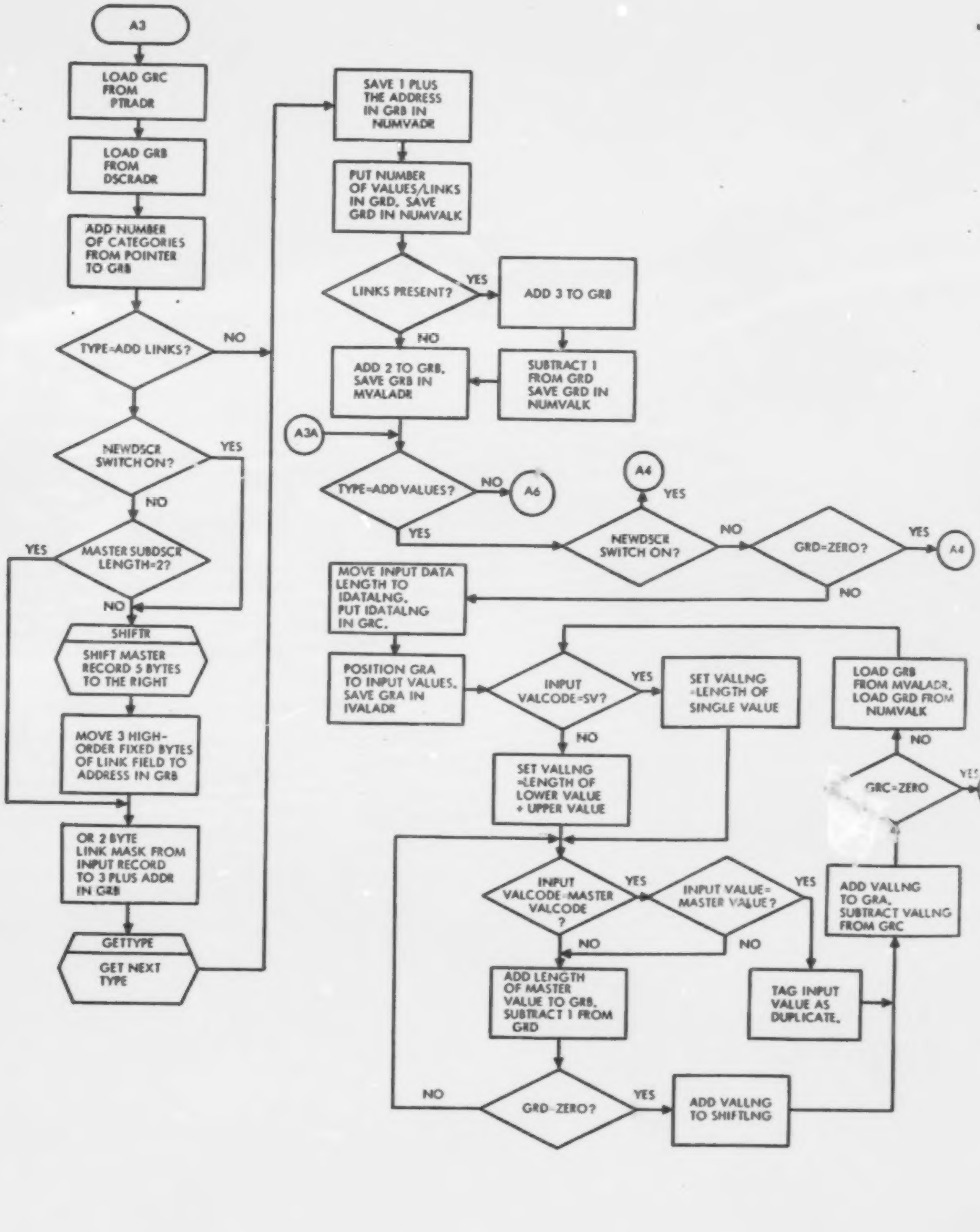

MAINT1
OPEN INPUT AND OUTPUT TAPE FILES
LKMAST
READ DIRECTORIES FROM DATA CELL
LOAD XREF LOOK-UP PROG INTO MASTER REC BUFFER
WRITE X-REF LOOK-UP PROGRAM ON X-REF DISK
M2
GETTYPE
GET NEXT TYPE
TYPE=HDR
NO
YES
RESET ERRSW, REVSW, ENDSW, PUTSW.
TYPE = HDR ADD
NO
YES
READ X-REF LOOK-UP PROGRAM INTO MASTER REC BUFF
LKXREF
LOOK-UP DOC IN X-REF FILE
FOUND ?
YES
NO
X-REF OVER-RIDE ?
YES
NO
PRINT ERROR MESSAGE
WRITE X-REF LOOK-UP PROGRAM ON DISK
ADD 1 TO HIGHEST INTNO. MOVE INTNO TO DOCNO
BLDFXD
SET UP FIXED FIELD FOR THIS REC
BLDXRD
CREATE INPUT TO X-REF FILE UPDATE
SECURITY ?
NO
M5
YES
TURN ON SECURITY BIT IN FIXED FIELD OF REC
M5
LKMAST
LOOK-UP DOC IN MASTER FILE
FOUND ?
NO
PRINT ERROR MESSAGE
M2
YES
TYPE = HDR DEL
YES
M6
NO
GTXT
COPY ALL TEXT ON WORK FILE, BUILD TEXT TABLE
SECURITY ?
YES
MASTER REC SECURED ?
NO
SET SECURITY BIT IN MASTER RECORD
NO
MASTER REC SECURED
NO
PRINT ERROR MESSAGE
YES
TURN OFF SECURITY BIT IN MASTER REC
M5
BLDRTBL
BUILD RANK TABLE
MAINT2
GETTYPE
GET NEXT TYPE
TYPE=X'FF'
YES
M7
NO
TYPE=
ADD
A1
DEL
D1
REV
R1

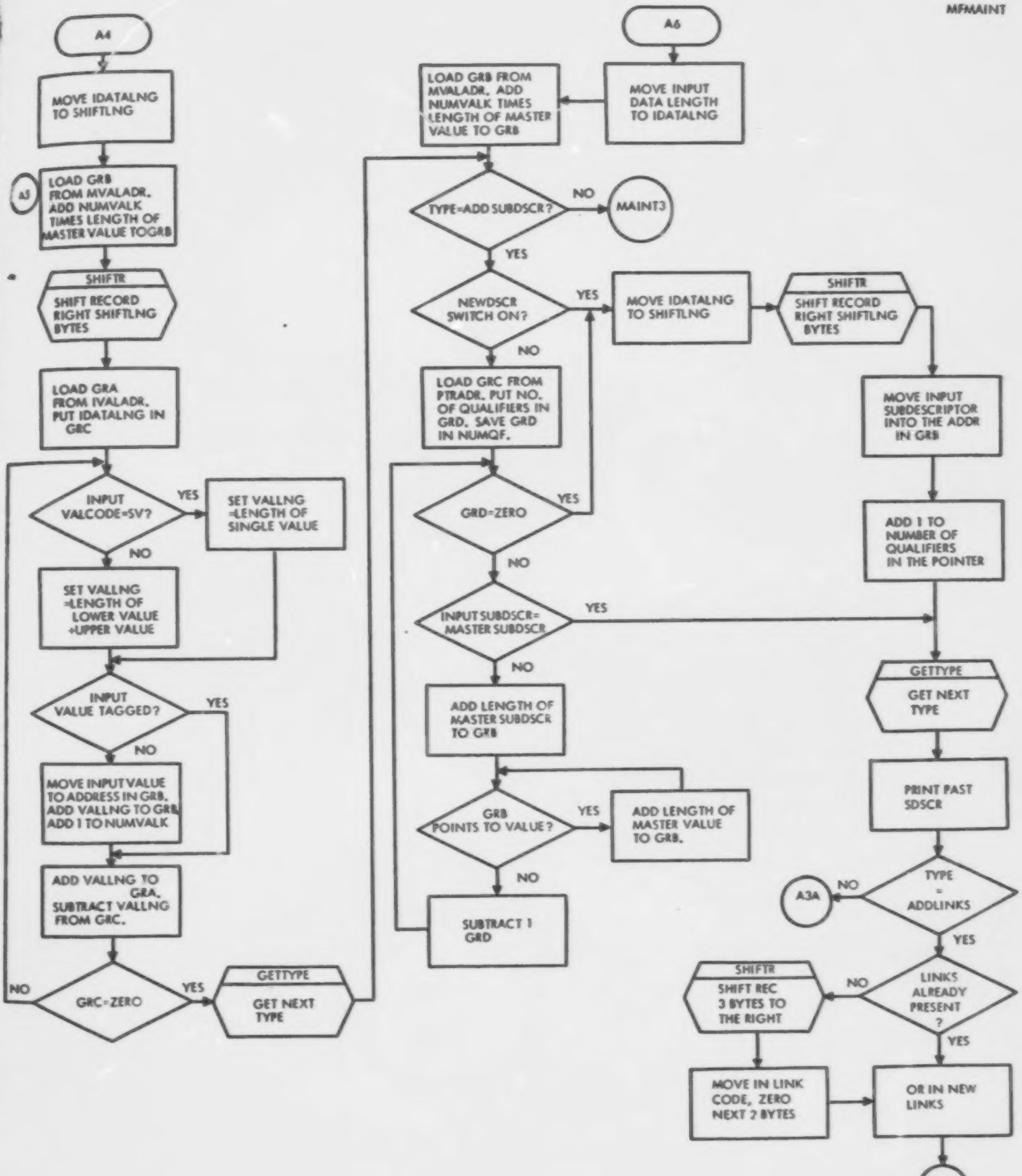

M6
BLDXRF
CREATE INPUT TO X-REF UPDATE
BLDRTBL
BUILD RANK TABLE
DELDOC
DELETE DOC FROM MASTER FILE
PUTPRNT
WRITE REC ON PRINT TAPE
BLDID
CREATE INPUT TO ID-FILE UPDATE
M2

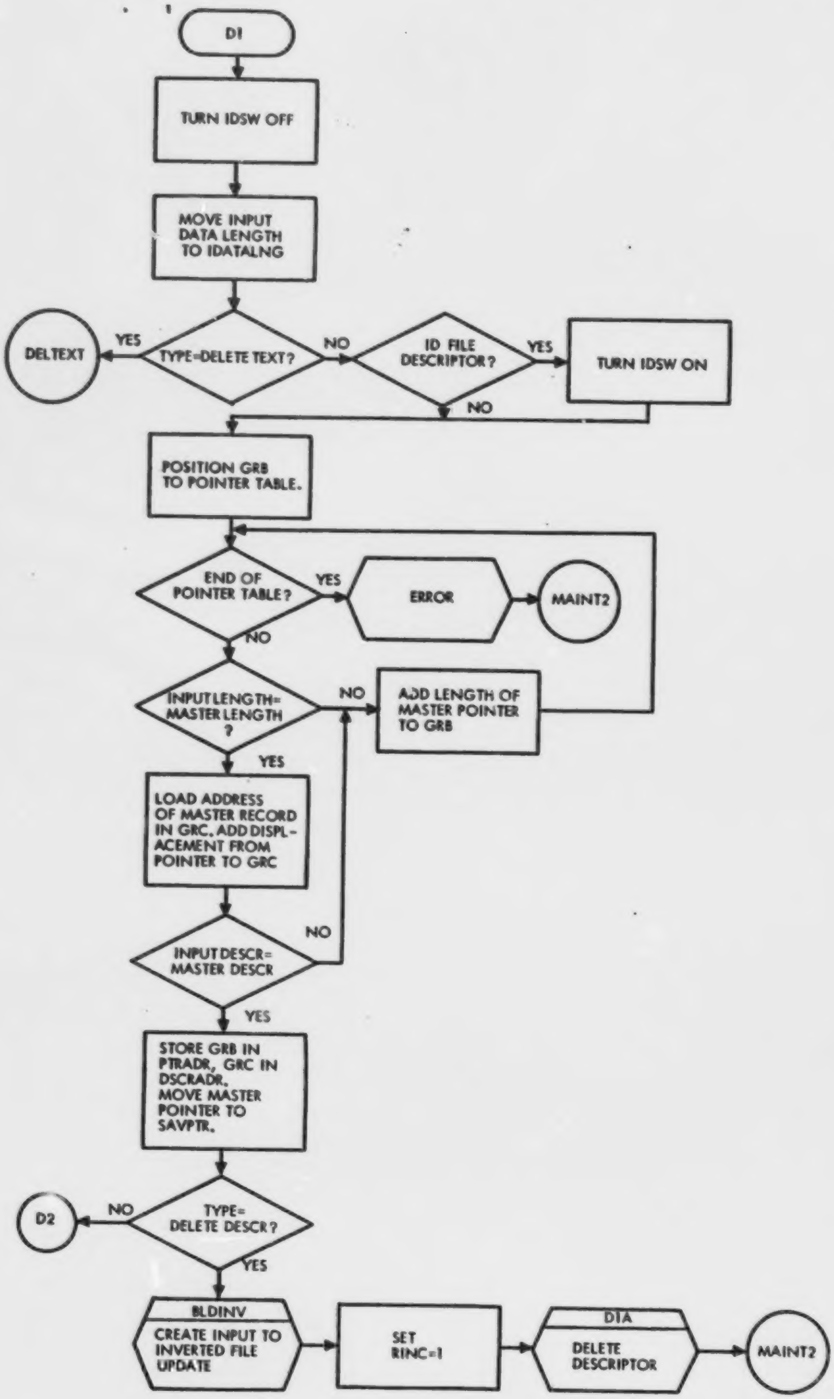

M7
ENDSW ON
NO
ENDDOC
END OF DOC PROCESSING
YES
WAS TEXT TABLE BUILT
NO
M2
YES
SET UP TO WRITE TEXT SEGMENTS
POINT TO TEXT TABLE
END OF TABLE
YES
M2
NO
IS THIS SEGMENT DELETED
YES
NO
READ THIS SEG FROM DISK
ENDDOC
WRITE SEG ON MASTER FILE
POINT TO NEXT ENTRY IN TABLE

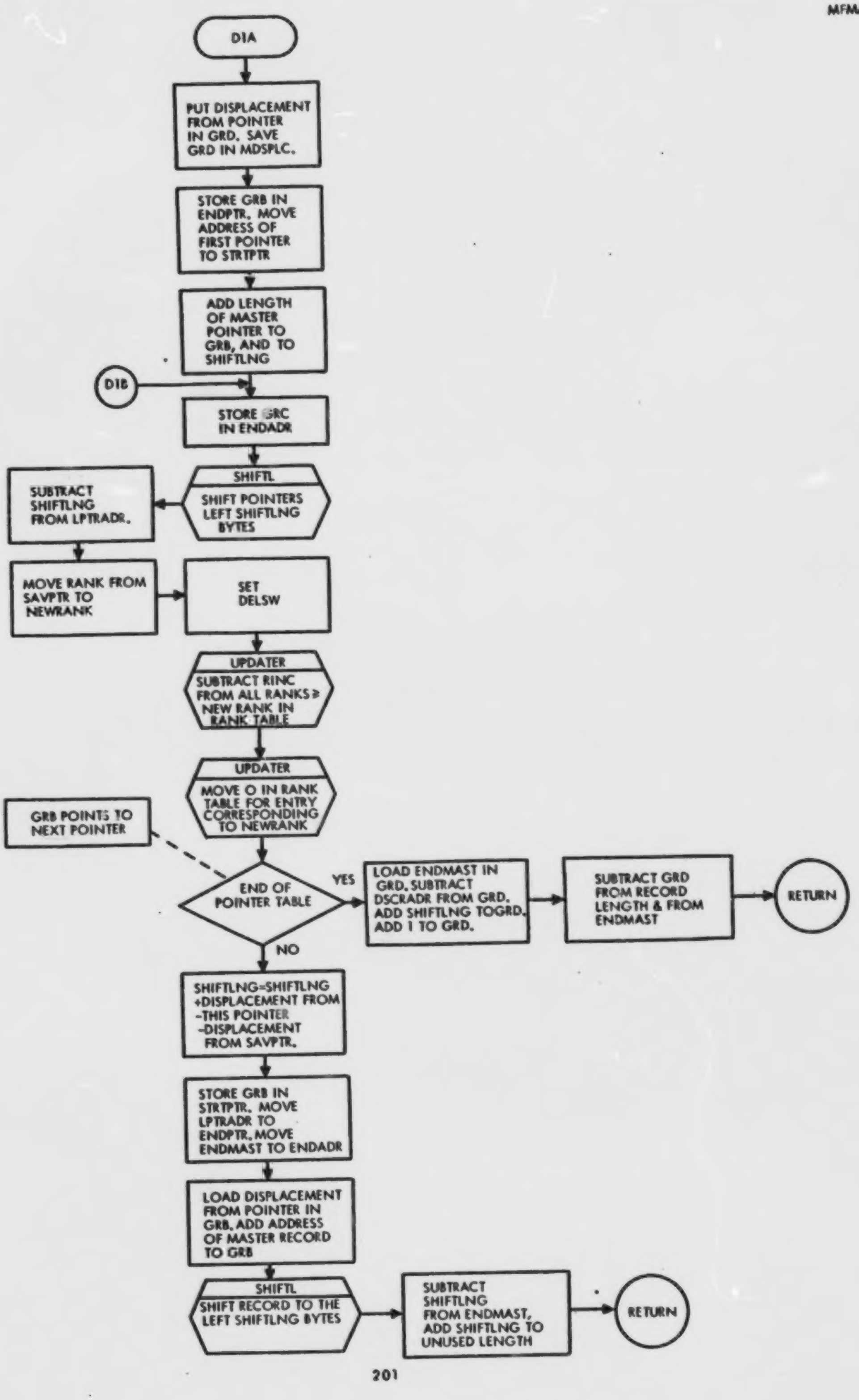

A1
TYPE=ADD TEXT?
YES
ADD-TEXT
NO
TYPE=ADD DSCR WITH RANK?
NO
MOVE HIGHESTR TO NEWRANK
YES
MOVE RANK FROM INPUT RECORD TO INPRANK
GETNEWR
MOVE RANK TABLE VALUE OF INPRANK TO NEWVAL
MOVE NEWVAL TO NEWRANK
POSITION GRB TO POINTER TABLE.
END OF POINTER TABLE?
YES
NO
POSITION GRA TO LENGTH OF INPUT DSCR
INPUT LENGTH= MASTER LENGTH?
NO
POSITION GRB TO NEXT POINTER
YES
LOAD ADDRESS OF MASTER RECORD IN GRC, ADD DISPLACEMENT FROM POINTER
INPUT DSCR: MASTER DSCR
LO
EQ
HI
STORE GRB IN PTRADR, STORE GRC IN DSCRADR
BLDINV
CREATE INPUT TO INVERTED FILE UPDATE
TURN NEWDSCR SWITCH ON
STORE GRB IN PTRADR
ADD LENGTH OF POINTER TO SHIFTLNG
POSITION GRB TO POINTER TABLE
END OF POINTER TABLE?
MOVE HIGHESTR TO NEWRANK, CHANGE TYPE TO ADD DSCR WITHOUT RANK
NO
NEWRANK =RANK IN POINTER?
NO
POSITION GRB TO NEXT POINTER
YES
TYPE=ADD DSCR WITH RANK?
NO
ADD 1 TO NEWRANK
ZERO GRB, PUT LENGTH OF DESCRIPTOR IN GRB.
YES
ADD ADDRESS OF MASTER RECORD TO GRB
ADD DISPLACEMENT FROM POINTER TO GRB, ADD LENGTH OF POINTER TO GRB
TYPE=ADD DSCR WITH RANK?
NO
YES
ADD LENGTH OF INPUT DESCRIPTOR TO SHIFTLNG, STORE GRB IN DSCRADR
SHIFTR
SHIFT RECORD TO THE RIGHT SHIFTLNG BYTES
UPDATER
ADD 1 TO ALL RANKS ≥NEWRANK IN RANK TABLE. MOVE NEWRANK TO NEXTRANK TABLE ENTRY
MOVE INPUT DESCR TO ADDRESS IN GRB. SUBTRACT ADDRESS OF MASTER REC FROM GRB. STORE GRB IN DSPLCMNT.
LOAD GRB FROM PTRADR, MOVE LENGTH OF POINTER TO SHIFTLNG
SET NEWRANK =0 SET END ADDRESS FOR POINTERS
SHIFTR
SHIFT POINTERS TO THE RIGHT SHFTLNG BYTES
MOVE SKELETON POINTER TO ADDRESS IN GRC ADD LENGTH OF MASTER POINTER TO LPTRADR
GETTYPE
GET NEXT TYPE
A2

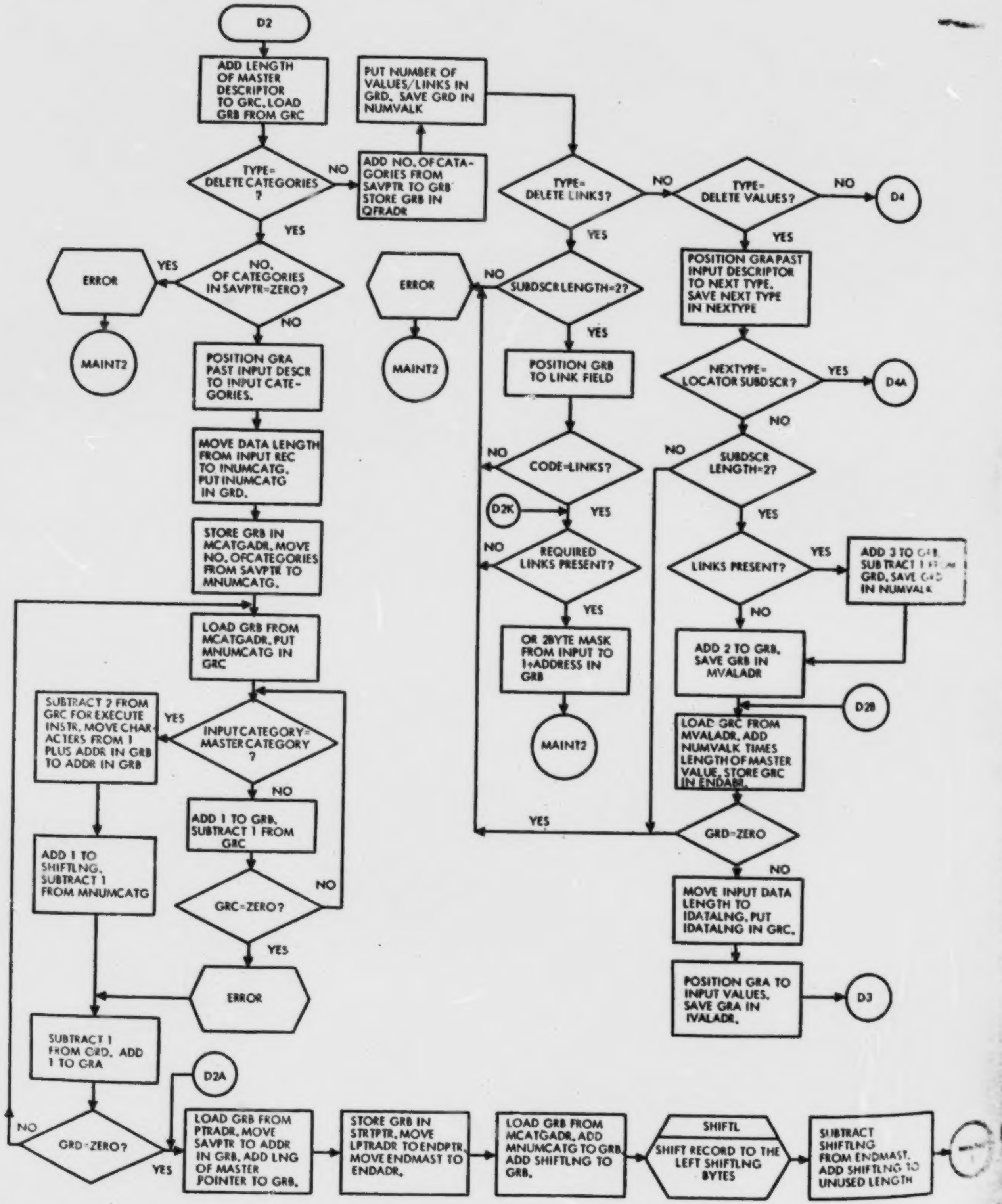

A2
ID-FILE DESCRIPTOR?
NO
YES
TURN IDSW ON
TYPE=ADD CATEGORIES ?
NO
A3
YES
MOVE DATA LENGTH FROM INPUT REC TO INUMCATG. PUT INUMCATG IN GRD.
POSITION GRA TO INPUT CATEGORIES. SAVE GRA IN ICATGADR.
MOVE NUMBER OF CATEGORIES FROM MASTER RECORD TO MNUMCATG.
ADD LENGTH OF MASTER DSCR TO GRB TO POINT TO CATEGORIES IN MASTER REC.
SAVE GRB IN MCATGADR
MNUMCATG=O
YES
NO
STORE GRD IN SHIFTLNG
SHIFTR
SHIFT RIGHT
LOAD GRB FROM MCATGADR. PUT MNUMCATG IN GRC.
INPUT CATEGORY MASTER CATEGORY ?
YES
NO
ADD 1 TO GRB. SUBTRACT 1 FROM GRC.
GRC=ZERO?
NO
YES
ADD 1 TO SHIFTLNG. SUBTRACT 1 FROM GRD.
GRD=ZERO?
YES
NO
ADD 1 TO GRA TO POINT TO NEXT INPUT CATEGORY.
MOVE INPUT CATEGORIES INTO REC
RELATED TERM?
NO
YES
LINKS WITH MAIN TERM
NO
YES
SET TYPE =ADDLINKS
A3
SHIFTR
SHIFT MASTER RECORD TO THE RIGHT SHIFTLNG BYTES
LOAD GRA FROM ICATGADR. PUT INUMCATG IN GRD
LOAD GRB FROM MCATGADR. PUT MNUMCATG IN GRC
INPUT CATEGORY: MASTER CATEGORY
EQ
HI
LO
ADD 1 TO GRB SUBTRACT 1 FROM GRC
GRC=ZERO?
YES
NO
MOVE CATEGORIES RIGHT 1 BYTE USING ADDRESS IN GRB & COUNT IN GRC
MOVE INPUT CATEGORY FROM ADDRESS IN GRA TO ADDRESS IN GRB
ADD 1 TO GRA. SUBTRACT 1 FROM GRD. ADD 1 TO MNUMCATG
ADD 1 TO NUMBER OF CATEGORIES IN THE POINTER
GRD=ZERO?
NO
YES
GETTYPE
GET NEXT TYPE

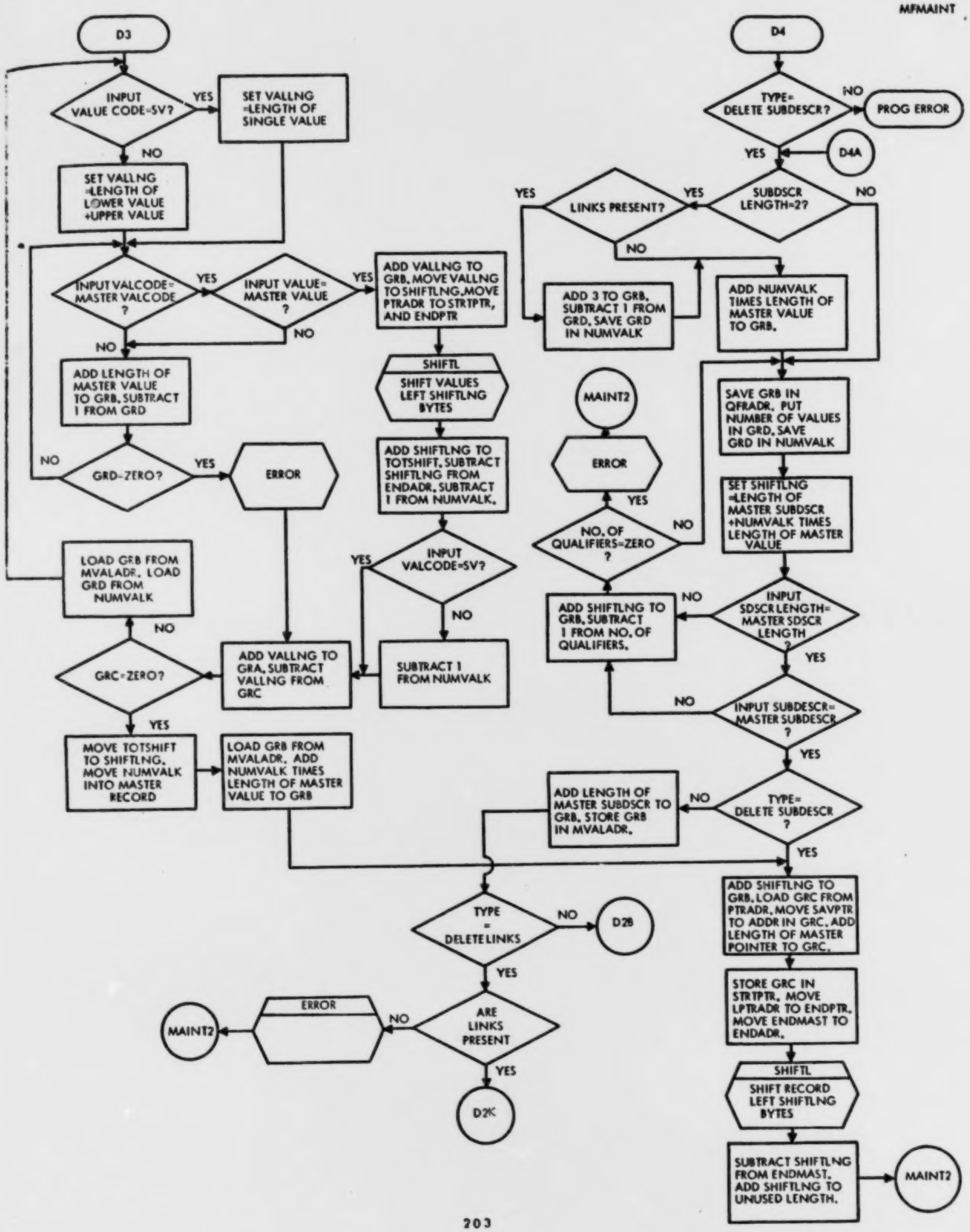

A3
LOAD GRC FROM PTRADR
LOAD GRB FROM DSCRADR
ADD NUMBER OF CATEGORIES FROM POINTER TO GRB
TYPE=ADD LINKS?
NO
NEWDSCR SWITCH ON?
YES
NO
MASTER SUBDSCR LENGTH=2?
YES
NO
SHIFTR
SHIFT MASTER RECORD 5 BYTES TO THE RIGHT
MOVE 3 HIGH-ORDER FIXED BYTES OF LINK FIELD TO ADDRESS IN GRB
OR 2 BYTE LINK MASK FROM INPUT RECORD TO 3 PLUS ADDR IN GRB
GETTYPE
GET NEXT TYPE
SAVE 1 PLUS THE ADDRESS IN GRB IN NUMVADR
PUT NUMBER OF VALUES/LINKS IN GRD, SAVE GRD IN NUMVALK
LINKS PRESENT?
YES
ADD 3 TO GRB
NO
ADD 2 TO GRB, SAVE GRB IN MVALADR
SUBTRACT 1 FROM GRD SAVE GRD IN NUMVALK
A3A
TYPE=ADD VALUES?
NO
A6
YES
A4
YES
NEWDSCR SWITCH ON?
NO
GRD=ZERO?
YES
A4
NO
MOVE INPUT DATA LENGTH TO IDATALNG, PUT IDATALNG IN GRC.
POSITION GRA TO INPUT VALUES, SAVE GRA IN IVALADR
INPUT VALCODE=SV?
YES
SET VALLNG =LENGTH OF SINGLE VALUE
NO
SET VALLNG =LENGTH OF LOWER VALUE + UPPER VALUE
LOAD GRB FROM MVALADR, LOAD GRD FROM NUMVALK
NO
GRC=ZERO
YES
INPUT VALCODE=MASTER VALCODE ?
YES
INPUT VALUE= MASTER VALUE?
YES
NO
NO
ADD LENGTH OF MASTER VALUE TO GRB, SUBTRACT 1 FROM GRD
TAG INPUT VALUE AS DUPLICATE.
ADD VALLNG TO GRA, SUBTRACT VALLNG FROM GRC
NO
GRD=ZERO?
YES
ADD VALLNG TO SHIFTLNG

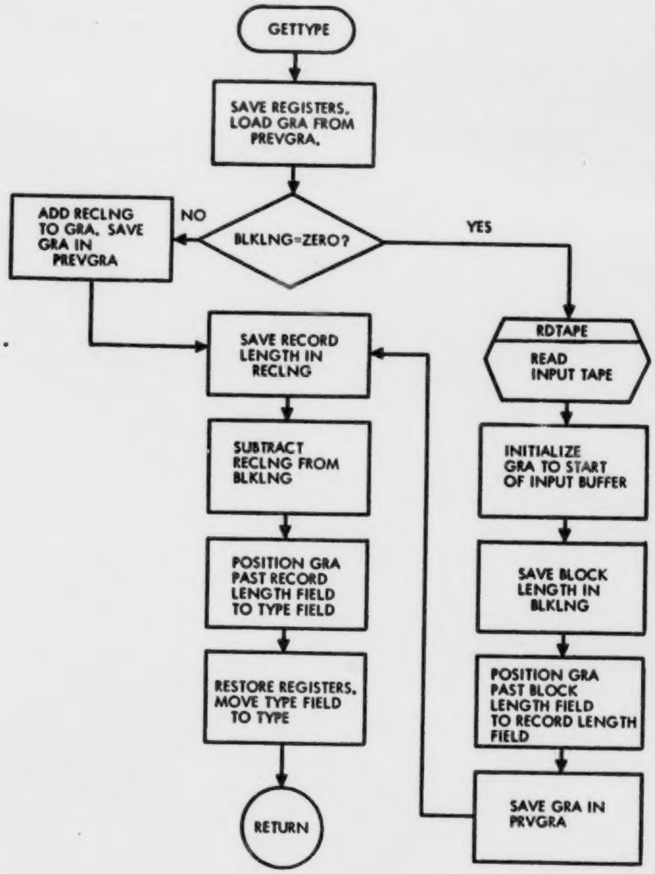

A4
MOVE IDATALNG TO SHIFTLNG
A5
LOAD GRB FROM MVALADR. ADD NUMVALK TIMES LENGTH OF MASTER VALUE TOGRB
SHIFTR
SHIFT RECORD RIGHT SHIFTLNG BYTES
LOAD GRA FROM IVALADR. PUT IDATALNG IN GRC
INPUT VALCODE=SV?
YES
SET VALLNG =LENGTH OF SINGLE VALUE
NO
SET VALLNG =LENGTH OF LOWER VALUE +UPPER VALUE
INPUT VALUE TAGGED?
YES
NO
MOVE INPUT VALUE TO ADDRESS IN GRB. ADD VALLNG TO GRB ADD 1 TO NUMVALK
ADD VALLNG TO GRA. SUBTRACT VALLNG FROM GRC.
NO
GRC=ZERO
YES
GETTYPE
GET NEXT TYPE
A6
MOVE INPUT DATA LENGTH TO IDATALNG
LOAD GRB FROM MVALADR. ADD NUMVALK TIMES LENGTH OF MASTER VALUE TO GRB
TYPE=ADD SUBDSCR?
NO
MAINT3
YES
NEWDSCR SWITCH ON?
YES
MOVE IDATALNG TO SHIFTLNG
SHIFTR
SHIFT RECORD RIGHT SHIFTLNG BYTES
NO
LOAD GRC FROM PTRADR. PUT NO. OF QUALIFIERS IN GRD. SAVE GRD IN NUMQF.
MOVE INPUT SUBDESCRIPTOR INTO THE ADDR IN GRB
GRD=ZERO
YES
NO
ADD 1 TO NUMBER OF QUALIFIERS IN THE POINTER
INPUT SUBDSCR= MASTER SUBDSCR
YES
NO
ADD LENGTH OF MASTER SUBDSCR TO GRB
GETTYPE
GET NEXT TYPE
PRINT PAST SDSCR
GRB POINTS TO VALUE?
YES
ADD LENGTH OF MASTER VALUE TO GRB.
NO
SUBTRACT 1 GRD
A3A
NO
TYPE = ADDLINKS
YES
SHIFTR
SHIFT REC 3 BYTES TO THE RIGHT
NO
LINKS ALREADY PRESENT ?
YES
MOVE IN LINK CODE, ZERO NEXT 2 BYTES
OR IN NEW LINKS
A3A

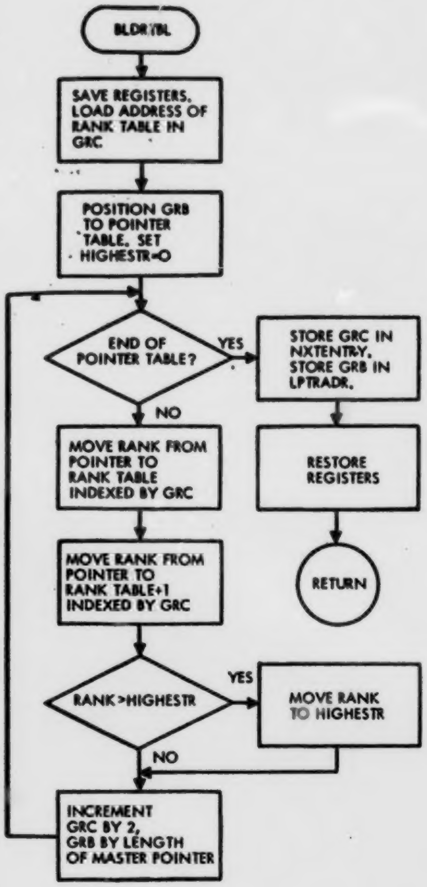

D1
TURN IDSW OFF
MOVE INPUT DATA LENGTH TO IDATALNG
TYPE=DELETE TEXT?
YES
DELTEXT
NO
ID FILE DESCRIPTOR?
YES
TURN IDSW ON
NO
POSITION GRB TO POINTER TABLE.
END OF POINTER TABLE?
YES
ERROR
MAINT2
NO
INPUT LENGTH= MASTER LENGTH ?
NO
ADD LENGTH OF MASTER POINTER TO GRB
YES
LOAD ADDRESS OF MASTER RECORD IN GRC. ADD DISPL- ACEMENT FROM POINTER TO GRC
INPUT DESCR= MASTER DESCR
NO
YES
STORE GRB IN PTRADR, GRC IN DSCRADR. MOVE MASTER POINTER TO SAVPTR.
TYPE= DELETE DESCR?
NO
D2
YES
BLDINV
CREATE INPUT TO INVERTED FILE UPDATE
SET RINC=1
DIA
DELETE DESCRIPTOR
MAINT2

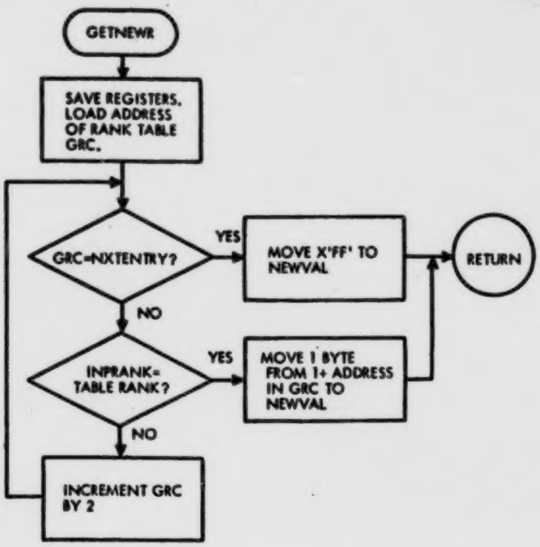

DIA
PUT DISPLACEMENT FROM POINTER IN GRD. SAVE GRD IN MDSPLC.
STORE GRB IN ENDPTR. MOVE ADDRESS OF FIRST POINTER TO STRTPTR
ADD LENGTH OF MASTER POINTER TO GRB, AND TO SHIFTLNG
DIB
STORE GRC IN ENDADR
SHIFTL
SHIFT POINTERS LEFT SHIFTLNG BYTES
SUBTRACT SHIFTLNG FROM LPTRADR.
MOVE RANK FROM SAVPTR TO NEWRANK
SET DELSW
UPDATER
SUBTRACT RINC FROM ALL RANKS ≥ NEW RANK IN RANK TABLE
UPDATER
MOVE 0 IN RANK TABLE FOR ENTRY CORRESPONDING TO NEWRANK
GRB POINTS TO NEXT POINTER
END OF POINTER TABLE
YES
LOAD ENDMAST IN GRD. SUBTRACT DSCRADR FROM GRD. ADD SHIFTLNG TOGRD. ADD 1 TO GRD.
SUBTRACT GRD FROM RECORD LENGTH & FROM ENDMAST
RETURN
NO
SHIFTLNG=SHIFTLNG +DISPLACEMENT FROM -THIS POINTER -DISPLACEMENT FROM SAVPTR.
STORE GRB IN STRTPTR. MOVE LPTRADR TO ENDPTR. MOVE ENDMAST TO ENDADR
LOAD DISPLACEMENT FROM POINTER IN GRB. ADD ADDRESS OF MASTER RECORD TO GRB
SHIFTL
SHIFT RECORD TO THE LEFT SHIFTLNG BYTES
SUBTRACT SHIFTLNG FROM ENDMAST, ADD SHIFTLNG TO UNUSED LENGTH
RETURN

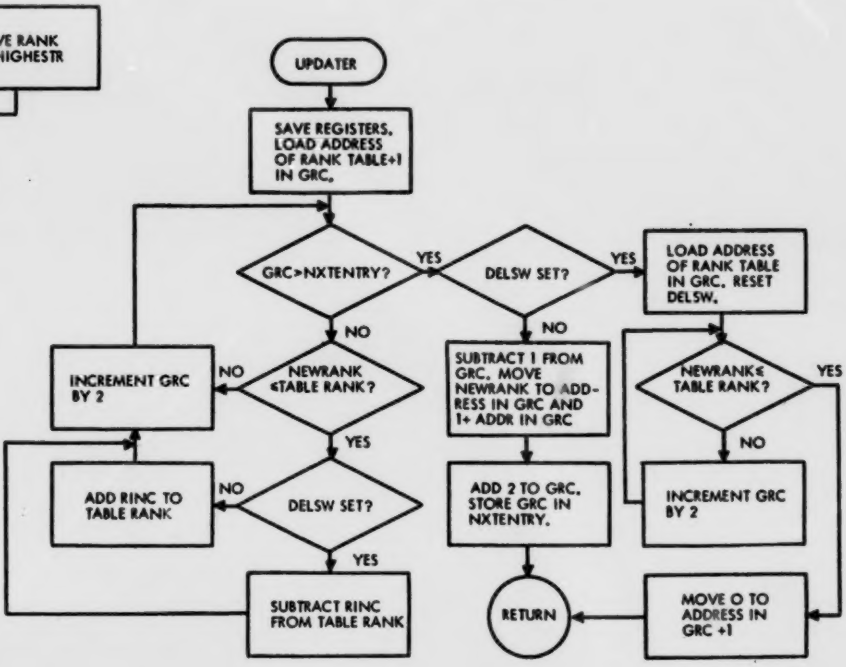

D2
ADD LENGTH OF MASTER DESCRIPTOR TO GRC, LOAD GRB FROM GRC
TYPE= DELETE CATEGORIES ?
NO
YES
ADD NO. OF CATAGORIES FROM SAVPTR TO GRB STORE GRB IN QFRADR
PUT NUMBER OF VALUES/LINKS IN GRD, SAVE GRD IN NUMVALK
TYPE= DELETE LINKS?
NO
YES
TYPE= DELETE VALUES?
NO
D4
YES
ERROR
YES
NO. OF CATEGORIES IN SAVPTR=ZERO?
NO
MAINT2
ERROR
NO
SUBDSCR LENGTH=2?
YES
MAINT2
POSITION GRA PAST INPUT DESCRIPTOR TO NEXT TYPE, SAVE NEXT TYPE IN NEXTYPE
NEXTYPE= LOCATOR SUBDSCR?
YES
D4A
NO
POSITION GRA PAST INPUT DESCR TO INPUT CATEGORIES.
POSITION GRB TO LINK FIELD
MOVE DATA LENGTH FROM INPUT REC TO INUMCATG, PUT INUMCATG IN GRD.
NO
CODE=LINKS?
NO
SUBDSCR LENGTH=2?
YES
D2K
YES
STORE GRB IN MCATGADR, MOVE NO. OFCATEGORIES FROM SAVPTR TO MNUMCATG.
NO
REQUIRED LINKS PRESENT?
YES
LINKS PRESENT?
YES
ADD 3 TO GRB, SUBTRACT 1 FROM GRD, SAVE GRD IN NUMVALK
NO
LOAD GRB FROM MCATGADR, PUT MNUMCATG IN GRC
OR 2BYTE MASK FROM INPUT TO 1+ADDRESS IN GRB
ADD 2 TO GRB, SAVE GRB IN MVALADR
MAINT2
D2B
SUBTRACT 2 FROM GRC FOR EXECUTE INSTR, MOVE CHARACTERS FROM 1 PLUS ADDR IN GRB TO ADDR IN GRB
YES
INPUT CATEGORY= MASTER CATEGORY ?
NO
LOAD GRC FROM MVALADR, ADD NUMVALK TIMES LENGTH OF MASTER VALUE, STORE GRC IN ENDADR.
ADD 1 TO GRB, SUBTRACT 1 FROM GRC
YES
GRD=ZERO
NO
ADD 1 TO SHIFTLNG, SUBTRACT 1 FROM MNUMCATG
NO
GRC=ZERO?
YES
MOVE INPUT DATA LENGTH TO IDATALNG, PUT IDATALNG IN GRC.
ERROR
POSITION GRA TO INPUT VALUES, SAVE GRA IN IVALADR.
D3
SUBTRACT 1 FROM GRD, ADD 1 TO GRA
D2A
NO
GRD=ZERO?
YES
LOAD GRB FROM PTRADR, MOVE SAVPTR TO ADDR IN GRB, ADD LNG OF MASTER POINTER TO GRB.
STORE GRB IN STRTPTR, MOVE LPTRADR TO ENDPTR, MOVE ENDMAST TO ENDADR.
LOAD GRB FROM MCATGADR, ADD MNUMCATG TO GRB, ADD SHIFTLNG TO GRB.
SHIFTL
SHIFT RECORD TO THE LEFT SHIFTLNG BYTES
SUBTRACT SHIFTLNG FROM ENDMAST, ADD SHIFTLNG TO UNUSED LENGTH

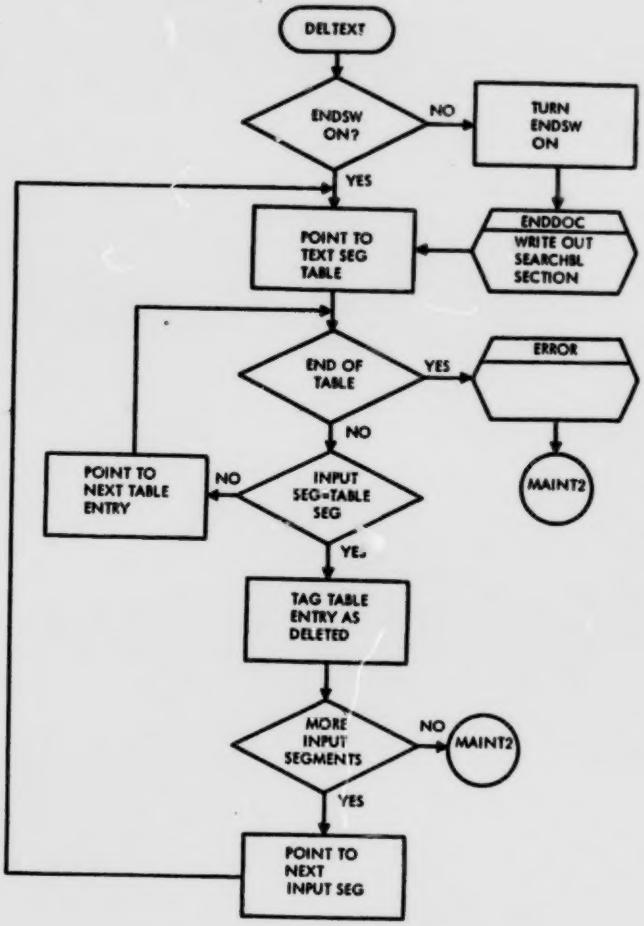

D3
INPUT VALUE CODE=SV?
YES
SET VALLNG =LENGTH OF SINGLE VALUE
NO
SET VALLNG =LENGTH OF LOWER VALUE +UPPER VALUE
INPUT VALCODE= MASTER VALCODE ?
YES
INPUT VALUE= MASTER VALUE ?
YES
ADD VALLNG TO GRB, MOVE VALLNG TO SHIFTLNG, MOVE PTRADR TO STRTPTR, AND ENDPTR
NO
NO
ADD LENGTH OF MASTER VALUE TO GRB, SUBTRACT 1 FROM GRD
SHIFTL
SHIFT VALUES LEFT SHIFTLNG BYTES
NO
GRD=ZERO?
YES
ERROR
ADD SHIFTLNG TO TOTSHIFT, SUBTRACT SHIFTLNG FROM ENDADR, SUBTRACT 1 FROM NUMVALK.
LOAD GRB FROM MVALADR, LOAD GRD FROM NUMVALK
YES
INPUT VALCODE=SV?
NO
NO
GRC=ZERO?
ADD VALLNG TO GRA, SUBTRACT VALLNG FROM GRC
SUBTRACT 1 FROM NUMVALK
YES
MOVE TOTSHIFT TO SHIFTLNG, MOVE NUMVALK INTO MASTER RECORD
LOAD GRB FROM MVALADR, ADD NUMVALK TIMES LENGTH OF MASTER VALUE TO GRB
D4
TYPE= DELETE SUBDESCR?
NO
PROG ERROR
YES
D4A
YES
LINKS PRESENT?
YES
SUBDSCR LENGTH=2?
NO
NO
ADD 3 TO GRB, SUBTRACT 1 FROM GRD, SAVE GRD IN NUMVALK
ADD NUMVALK TIMES LENGTH OF MASTER VALUE TO GRB.
MAINT2
SAVE GRB IN QFRADR, PUT NUMBER OF VALUES IN GRD, SAVE GRD IN NUMVALK
ERROR
YES
NO
NO. OF QUALIFIERS=ZERO ?
SET SHIFTLNG =LENGTH OF MASTER SUBDSCR +NUMVALK TIMES LENGTH OF MASTER VALUE
NO
ADD SHIFTLNG TO GRB, SUBTRACT 1 FROM NO. OF QUALIFIERS.
INPUT SDSCR LENGTH= MASTER SDSCR LENGTH ?
YES
NO
INPUT SUBDESCR= MASTER SUBDESCR ?
YES
ADD LENGTH OF MASTER SUBDSCR TO GRB, STORE GRB IN MVALADR.
NO
TYPE= DELETE SUBDESCR ?
YES
ADD SHIFTLNG TO GRB, LOAD GRC FROM PTRADR, MOVE SAVPTR TO ADDR IN GRC, ADD LENGTH OF MASTER POINTER TO GRC.
TYPE = DELETE LINKS
NO
D2B
YES
STORE GRC IN STRTPTR, MOVE LPTRADR TO ENDPTR, MOVE ENDMAST TO ENDADR.
MAINT2
ERROR
NO
ARE LINKS PRESENT
YES
SHIFTL
SHIFT RECORD LEFT SHIFTLNG BYTES
D2K
SUBTRACT SHIFTLNG FROM ENDMAST, ADD SHIFTLNG TO UNUSED LENGTH.
MAINT2

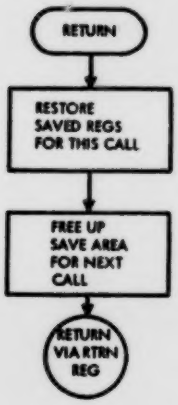

GETTYPE
SAVE REGISTERS. LOAD GRA FROM PREVGRA.
BLKLNG=ZERO?
NO
YES
ADD RECLNG TO GRA. SAVE GRA IN PREVGRA
RDTAPE
READ INPUT TAPE
SAVE RECORD LENGTH IN RECLNG
SUBTRACT RECLNG FROM BLKLNG
POSITION GRA PAST RECORD LENGTH FIELD TO TYPE FIELD
RESTORE REGISTERS. MOVE TYPE FIELD TO TYPE
RETURN
INITIALIZE GRA TO START OF INPUT BUFFER
SAVE BLOCK LENGTH IN BLKLNG
POSITION GRA PAST BLOCK LENGTH FIELD TO RECORD LENGTH FIELD
SAVE GRA IN PRVGRA

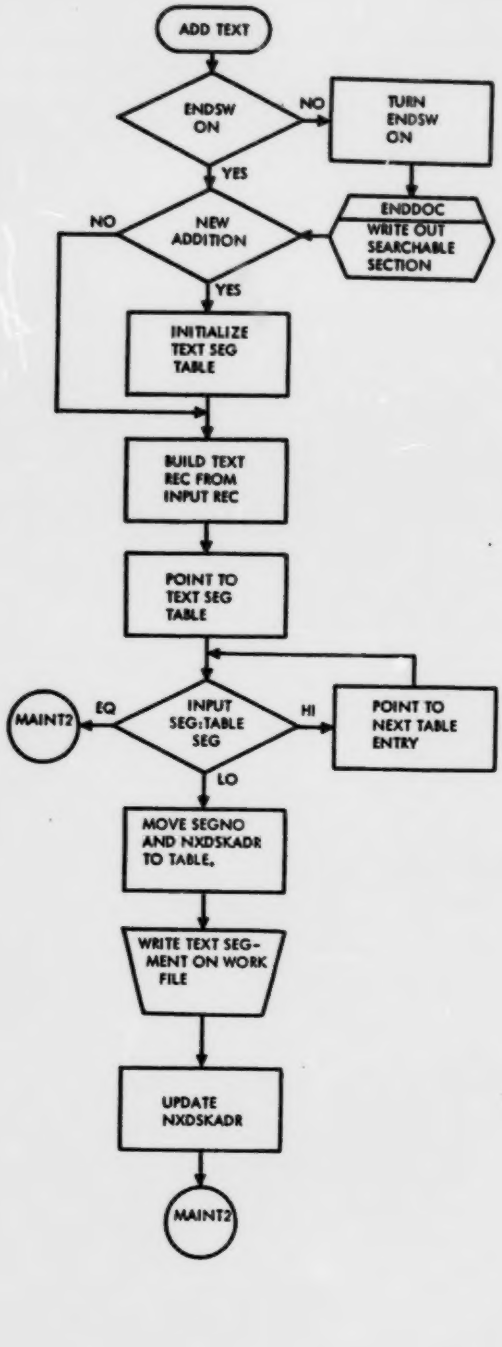

BLDRTBL
SAVE REGISTERS, LOAD ADDRESS OF RANK TABLE IN GRC
POSITION GRB TO POINTER TABLE, SET HIGHESTR=O
END OF POINTER TABLE?
YES
STORE GRC IN NXTENTRY, STORE GRB IN LPTRADR.
RESTORE REGISTERS
RETURN
NO
MOVE RANK FROM POINTER TO RANK TABLE INDEXED BY GRC
MOVE RANK FROM POINTER TO RANK TABLE+1 INDEXED BY GRC
RANK>HIGHESTR
YES
MOVE RANK TO HIGHESTR
NO
INCREMENT GRC BY 2, GRB BY LENGTH OF MASTER POINTER

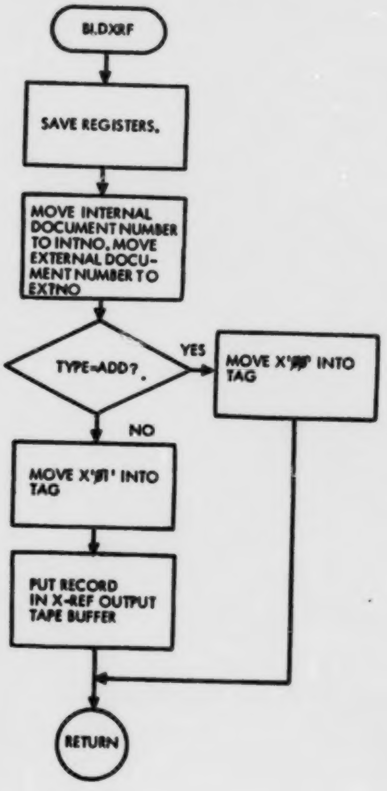

GETNEWR
SAVE REGISTERS, LOAD ADDRESS OF RANK TABLE GRC.
GRC=NXTENTRY?
YES
MOVE X'FF' TO NEWVAL
RETURN
NO
INPRANK= TABLE RANK?
YES
MOVE 1 BYTE FROM 1+ ADDRESS IN GRC TO NEWVAL
NO
INCREMENT GRC BY 2

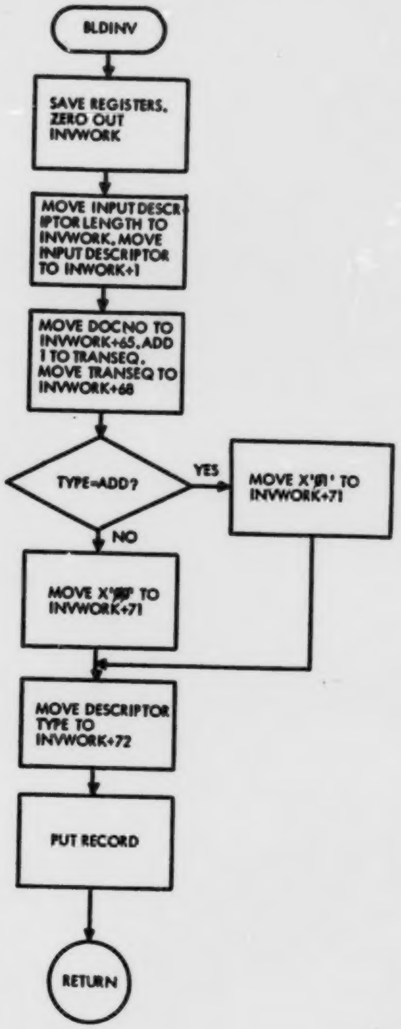

UPDATER
SAVE REGISTERS, LOAD ADDRESS OF RANK TABLE+1 IN GRC,
GRC>NXTENTRY?
YES
DELSW SET?
YES
LOAD ADDRESS OF RANK TABLE IN GRC, RESET DELSW,
NEWRANK≤ TABLE RANK?
YES
MOVE O TO ADDRESS IN GRC +1
NO
INCREMENT GRC BY 2
NO
SUBTRACT 1 FROM GRC, MOVE NEWRANK TO ADDRESS IN GRC AND 1+ ADDR IN GRC
ADD 2 TO GRC, STORE GRC IN NXTENTRY.
RETURN
NO
NEWRANK ≤TABLE RANK?
NO
INCREMENT GRC BY 2
YES
DELSW SET?
NO
ADD RINC TO TABLE RANK
YES
SUBTRACT RINC FROM TABLE RANK

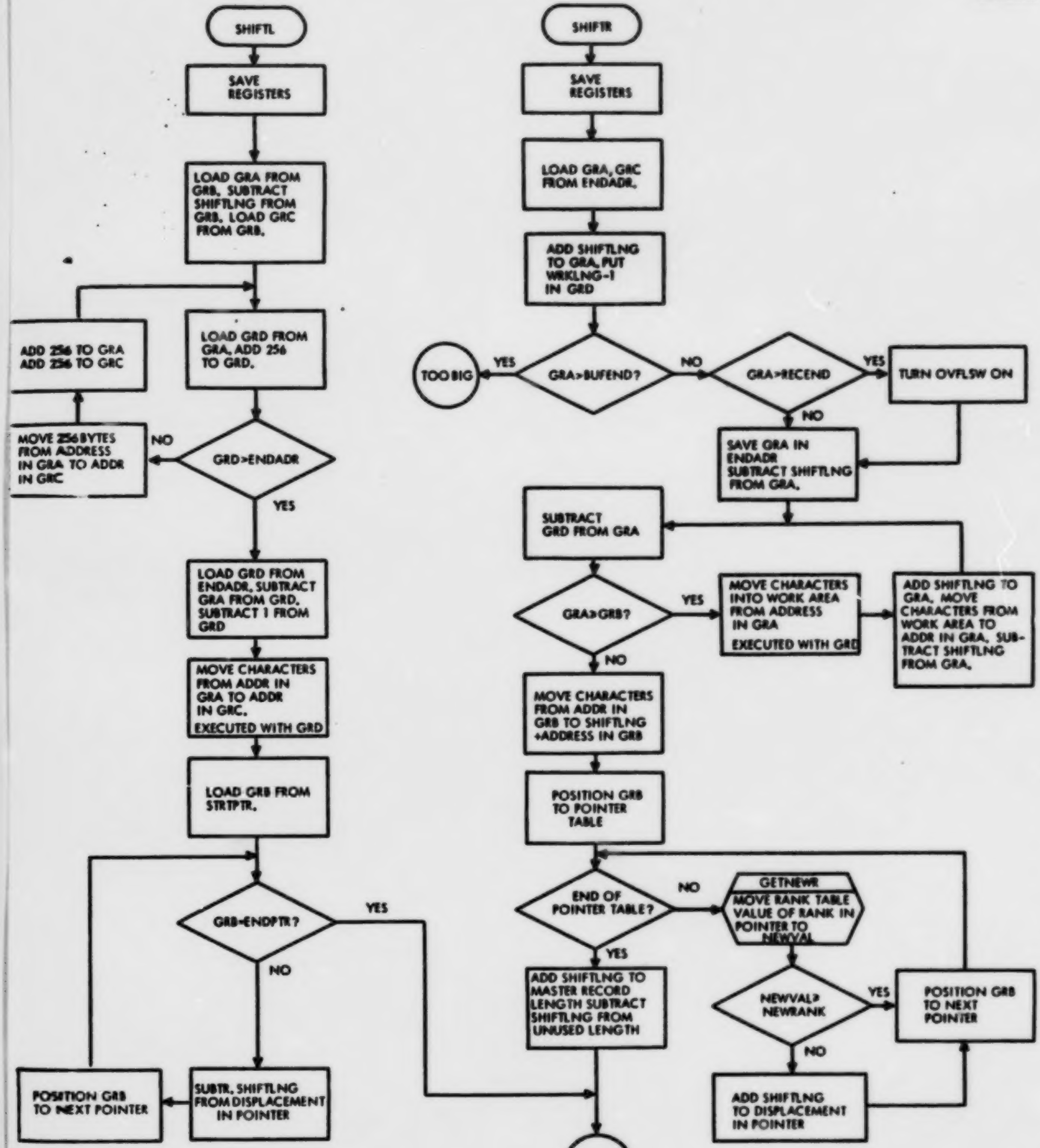

DELTEXT
ENDSW ON?
NO
TURN ENDSW ON
YES
ENDDOC
WRITE OUT SEARCHBL SECTION
POINT TO TEXT SEG TABLE
END OF TABLE
YES
ERROR
NO
MAINT2
POINT TO NEXT TABLE ENTRY
NO
INPUT SEG=TABLE SEG
YES
TAG TABLE ENTRY AS DELETED
MORE INPUT SEGMENTS
NO
MAINT2
YES
POINT TO NEXT INPUT SEG

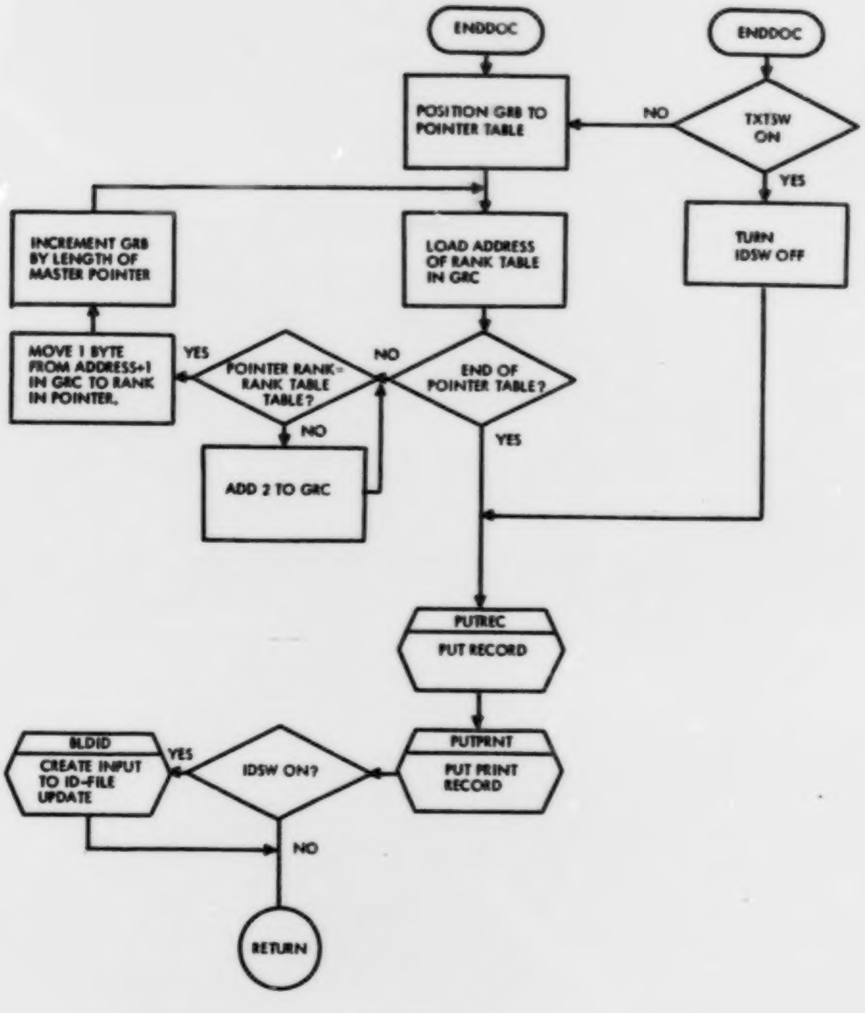

RETURN
RESTORE SAVED REGS FOR THIS CALL
FREE UP SAVE AREA FOR NEXT CALL
RETURN VIA RTRN REG

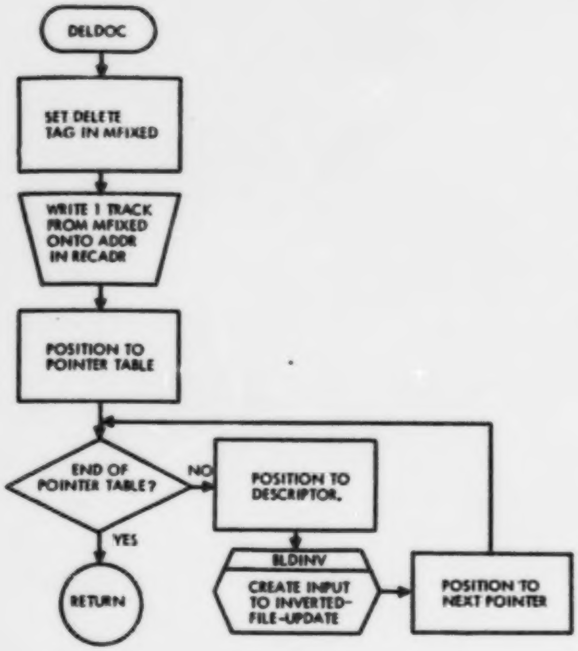

ADD TEXT
ENDSW ON
NO
TURN ENDSW ON
YES
ENDDOC
WRITE OUT SEARCHABLE SECTION
NO
NEW ADDITION
YES
INITIALIZE TEXT SEG TABLE
BUILD TEXT REC FROM INPUT REC
POINT TO TEXT SEG TABLE
MAINT2
EQ
INPUT SEG:TABLE SEG
HI
POINT TO NEXT TABLE ENTRY
LO
MOVE SEGNO AND NXDSKADR TO TABLE.
WRITE TEXT SEGMENT ON WORK FILE
UPDATE NXDSKADR
MAINT2

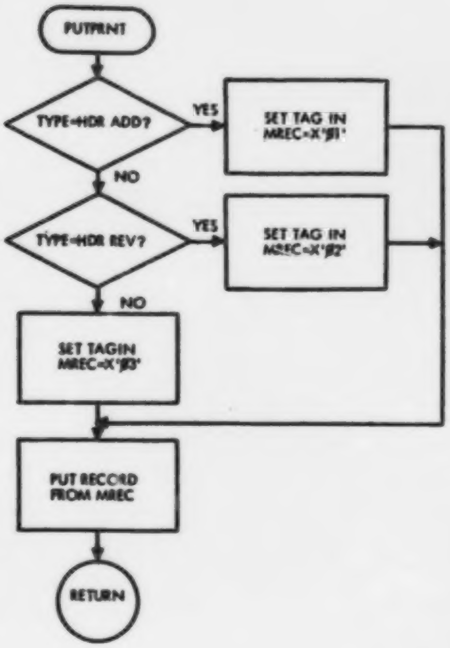

BLDXRF
SAVE REGISTERS.
MOVE INTERNAL DOCUMENT NUMBER TO INTNO. MOVE EXTERNAL DOCUMENT NUMBER TO EXTNO
TYPE=ADD?
YES
MOVE X'00' INTO TAG
NO
MOVE X'01' INTO TAG
PUT RECORD IN X-REF OUTPUT TAPE BUFFER
RETURN

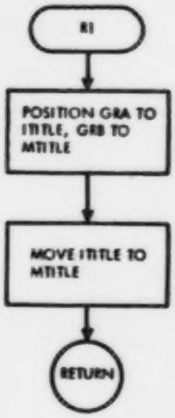

BLDINV
SAVE REGISTERS. ZERO OUT INVWORK
MOVE INPUT DESCRIPTOR LENGTH TO INVWORK. MOVE INPUT DESCRIPTOR TO INWORK+1
MOVE DOCNO TO INVWORK+65. ADD 1 TO TRANSEQ. MOVE TRANSEQ TO INVWORK+68
TYPE=ADD?
YES
MOVE X'01' TO INVWORK+71
NO
MOVE X'00' TO INVWORK+71
MOVE DESCRIPTOR TYPE TO INVWORK+72
PUT RECORD
RETURN

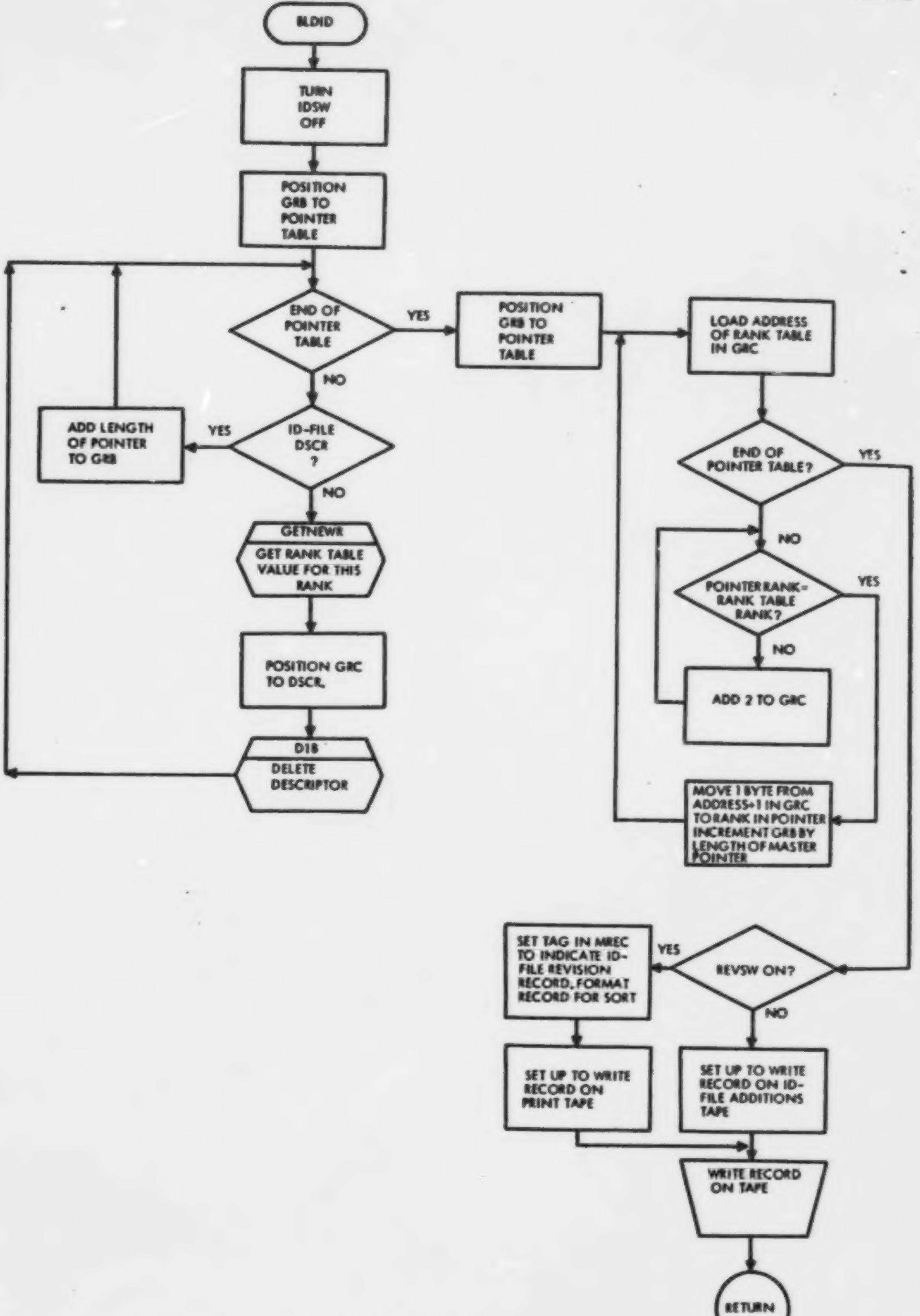

SHIFTL
SAVE REGISTERS
LOAD GRA FROM GRB, SUBTRACT SHIFTLNG FROM GRB, LOAD GRC FROM GRB.
LOAD GRD FROM GRA, ADD 256 TO GRD.
ADD 256 TO GRA ADD 256 TO GRC
MOVE 256 BYTES FROM ADDRESS IN GRA TO ADDR IN GRC
NO
GRD>ENDADR
YES
LOAD GRD FROM ENDADR, SUBTRACT GRA FROM GRD, SUBTRACT 1 FROM GRD
MOVE CHARACTERS FROM ADDR IN GRA TO ADDR IN GRC.
EXECUTED WITH GRD
LOAD GRB FROM STRTPTR.
GRB=ENDPTR?
YES
NO
SUBTR. SHIFTLNG FROM DISPLACEMENT IN POINTER
POSITION GRB TO NEXT POINTER
SHIFTR
SAVE REGISTERS
LOAD GRA, GRC FROM ENDADR.
ADD SHIFTLNG TO GRA, PUT WRKLNG-1 IN GRD
GRA>BUFEND?
YES
TOO BIG
NO
GRA>RECEND
YES
TURN OVFLSW ON
NO
SAVE GRA IN ENDADR SUBTRACT SHIFTLNG FROM GRA.
SUBTRACT GRD FROM GRA
GRA>GRB?
YES
MOVE CHARACTERS INTO WORK AREA FROM ADDRESS IN GRA
EXECUTED WITH GRD
ADD SHIFTLNG TO GRA, MOVE CHARACTERS FROM WORK AREA TO ADDR IN GRA, SUBTRACT SHIFTLNG FROM GRA.
NO
MOVE CHARACTERS FROM ADDR IN GRB TO SHIFTLNG +ADDRESS IN GRB
POSITION GRB TO POINTER TABLE
END OF POINTER TABLE?
NO
GETNEWR
MOVE RANK TABLE VALUE OF RANK IN POINTER TO NEWVAL
YES
ADD SHIFTLNG TO MASTER RECORD LENGTH SUBTRACT SHIFTLNG FROM UNUSED LENGTH
NEWVAL> NEWRANK
YES
POSITION GRB TO NEXT POINTER
NO
ADD SHIFTLNG TO DISPLACEMENT IN POINTER
RETURN

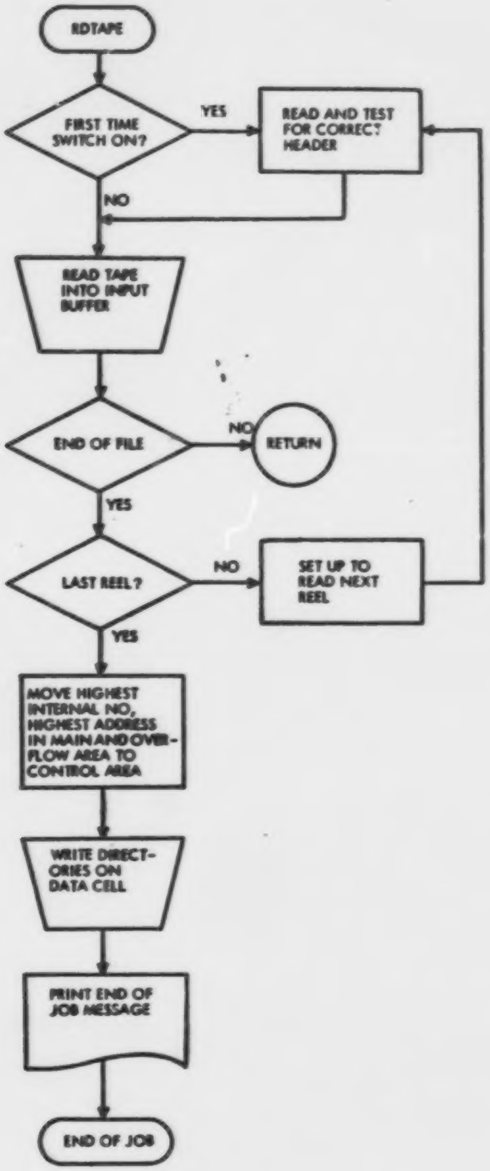

ENDDOC
ENDDOC
POSITION GRB TO POINTER TABLE
NO
TXTSW ON
YES
INCREMENT GRB BY LENGTH OF MASTER POINTER
LOAD ADDRESS OF RANK TABLE IN GRC
TURN IDSW OFF
MOVE 1 BYTE FROM ADDRESS+1 IN GRC TO RANK IN POINTER.
YES
POINTER RANK= RANK TABLE TABLE?
NO
END OF POINTER TABLE?
NO
ADD 2 TO GRC
YES
PUTREC
PUT RECORD
BLDID
CREATE INPUT TO ID-FILE UPDATE
YES
IDSW ON?
PUTPRNT
PUT PRINT RECORD
NO
RETURN

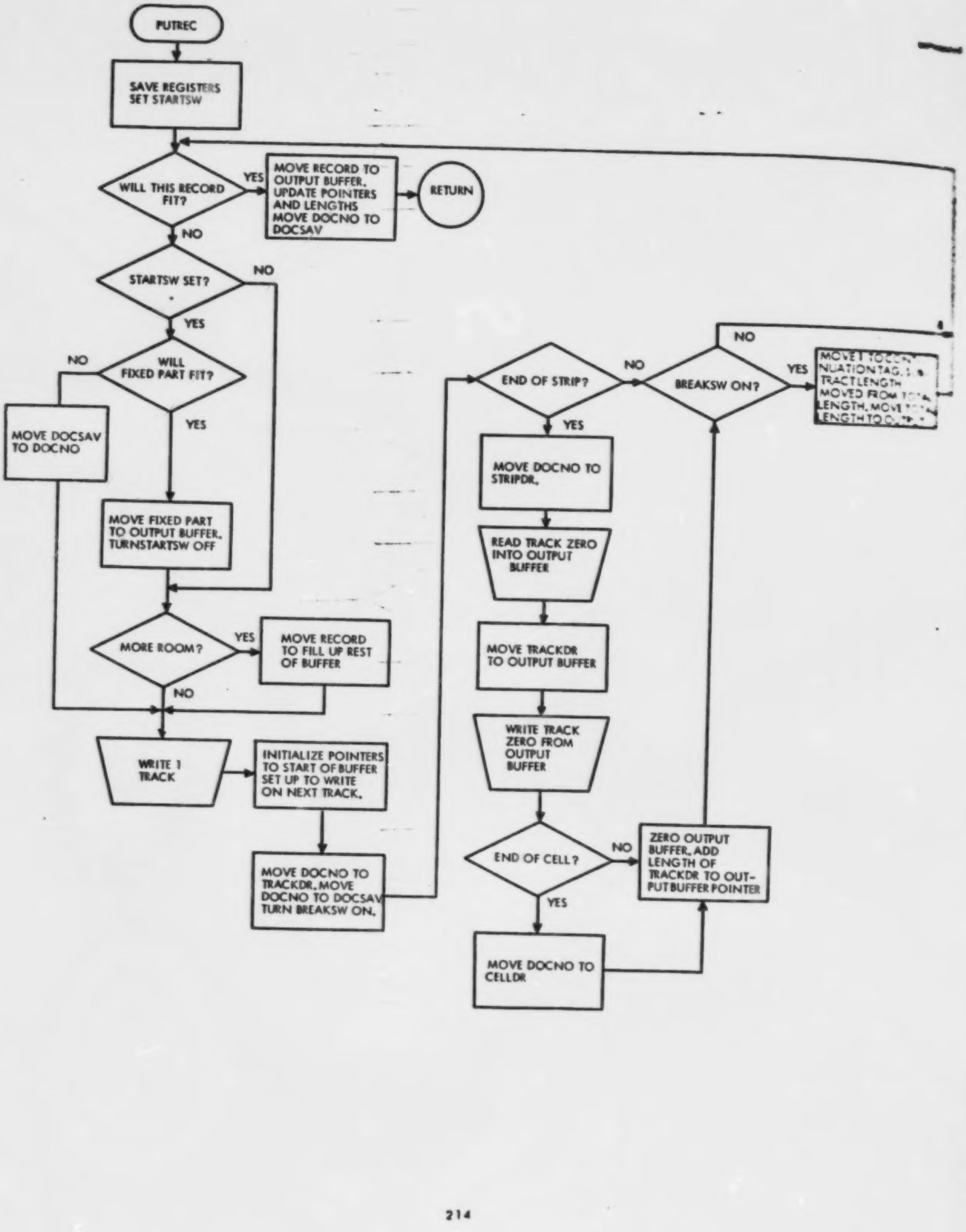

DELDOC
SET DELETE TAG IN MFIXED
WRITE 1 TRACK FROM MFIXED ONTO ADDR IN RECADR
POSITION TO POINTER TABLE
END OF POINTER TABLE?
NO
YES
POSITION TO DESCRIPTOR.
BLDINV
CREATE INPUT TO INVERTED-FILE-UPDATE
POSITION TO NEXT POINTER
RETURN

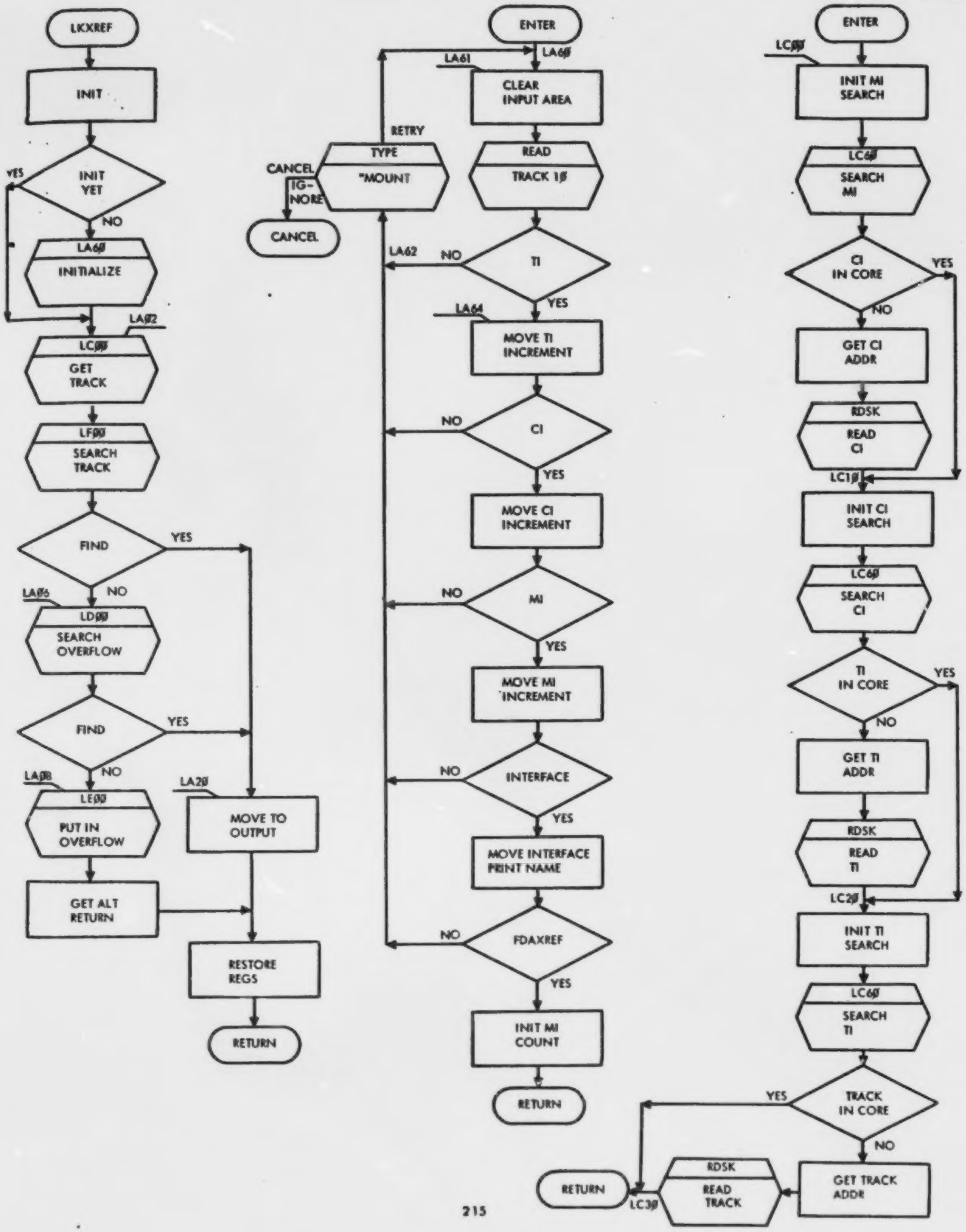

PUTPRINT
TYPE=HDR ADD?
YES
SET TAG IN MREC=X'01'
NO
TYPE=HDR REV?
YES
SET TAG IN MREC=X'02'
NO
SET TAGIN MREC=X'03'
PUT RECORD FROM MREC
RETURN


RI
POSITION GRA TO ITITLE, GRB TO MTITLE
MOVE ITITLE TO MTITLE
RETURN


BLDID
TURN IDSW OFF
POSITION GRB TO POINTER TABLE
END OF POINTER TABLE
YES
NO
POSITION GRB TO POINTER TABLE
LOAD ADDRESS OF RANK TABLE IN GRC
ID-FILE DSCR ?
YES
ADD LENGTH OF POINTER TO GRB
NO
GETNEWR
GET RANK TABLE VALUE FOR THIS RANK
POSITION GRC TO DSCR.
DIB
DELETE DESCRIPTOR
END OF POINTER TABLE?
YES
NO
POINTER RANK= RANK TABLE RANK?
YES
NO
ADD 2 TO GRC
MOVE 1 BYTE FROM ADDRESS+1 IN GRC TO RANK IN POINTER INCREMENT GRB BY LENGTH OF MASTER POINTER
REVSW ON?
YES
SET TAG IN MREC TO INDICATE ID-FILE REVISION RECORD, FORMAT RECORD FOR SORT
NO
SET UP TO WRITE RECORD ON PRINT TAPE
SET UP TO WRITE RECORD ON ID-FILE ADDITIONS TAPE
WRITE RECORD ON TAPE
RETURN


RDTAPE
FIRST TIME SWITCH ON?
YES
READ AND TEST FOR CORRECT HEADER
NO
READ TAPE INTO INPUT BUFFER
END OF FILE
NO
RETURN
YES
LAST REEL ?
NO
SET UP TO READ NEXT REEL
YES
MOVE HIGHEST INTERNAL NO, HIGHEST ADDRESS IN MAIN AND OVER-FLOW AREA TO CONTROL AREA
WRITE DIRECT-ORIES ON DATA CELL
PRINT END OF JOB MESSAGE
END OF JOB


PUTREC
SAVE REGISTERS SET STARTSW
WILL THIS RECORD FIT?
YES
MOVE RECORD TO OUTPUT BUFFER, UPDATE POINTERS AND LENGTHS MOVE DOCNO TO DOCSAV
RETURN
NO
STARTSW SET?
NO
YES
NO
WILL FIXED PART FIT?
YES
MOVE DOCSAV TO DOCNO
MOVE FIXED PART TO OUTPUT BUFFER, TURNSTARTSW OFF
MORE ROOM?
YES
MOVE RECORD TO FILL UP REST OF BUFFER
NO
WRITE 1 TRACK
INITIALIZE POINTERS TO START OF BUFFER SET UP TO WRITE ON NEXT TRACK.
MOVE DOCNO TO TRACKDR, MOVE DOCNO TO DOCSAV TURN BREAKSW ON.
END OF STRIP?
NO
BREAKSW ON?
NO
YES
MOVE 1 TO CONTINUATION TAG, SUBTRACT LENGTH MOVED FROM TOTAL LENGTH, MOVE TOTAL LENGTH TO OUTPUT
YES
MOVE DOCNO TO STRIPDR.
READ TRACK ZERO INTO OUTPUT BUFFER
MOVE TRACKDR TO OUTPUT BUFFER
WRITE TRACK ZERO FROM OUTPUT BUFFER
END OF CELL?
NO
ZERO OUTPUT BUFFER, ADD LENGTH OF TRACKDR TO OUTPUT BUFFER POINTER
YES
MOVE DOCNO TO CELLDR


LKXREF
INIT
INIT YET
YES
NO
LA60
INITIALIZE
LA02
LC00
GET TRACK
LF00
SEARCH TRACK
FIND
YES
NO
LA06
LD00
SEARCH OVERFLOW
FIND
YES
NO
LA08
LE00
PUT IN OVERFLOW
GET ALT RETURN
LA20
MOVE TO OUTPUT
RESTORE REGS
RETURN
ENTER
LA60
LA61
CLEAR INPUT AREA
RETRY
TYPE
"MOUNT
CANCEL
IG-NORE
CANCEL
READ
TRACK 10
LA62
NO
TI
YES
LA64
MOVE TI INCREMENT
NO
CI
YES
MOVE CI INCREMENT
NO
MI
YES
MOVE MI INCREMENT
NO
INTERFACE
YES
MOVE INTERFACE PRINT NAME
NO
FDAXREF
YES
INIT MI COUNT
RETURN
ENTER
LC00
INIT MI SEARCH
LC60
SEARCH MI
CI IN CORE
YES
NO
GET CI ADDR
RDSK
READ CI
LC10
INIT CI SEARCH
LC60
SEARCH CI
TI IN CORE
YES
NO
GET TI ADDR
RDSK
READ TI
LC20
INIT TI SEARCH
LC60
SEARCH TI
TRACK IN CORE
YES
NO
GET TRACK ADDR
RDSK
READ TRACK
LC30
RETURN

INVERTED FILE UPDATE PROGRAM (IVUPDT) PROCESSING LOGIC

During the initialization procedure all three tape files are opened, switches and counters are set, the index tables are readied for disk output and the first records on each input file are read.

1. Compare

   Compare Inverted File Update to Prior Inverted File Record.

2. Update Low

   If the update is low, create an inverted file record for this new term and read and process each additional update record for this term. Output the new inverted file record and return to the compare above.

3. Update High

   Scan the record for any block changes and change any Match as requested. Output this inverted file record and read the next inverted file input record and return to the compare above.

4. Update Equal

   If delete all, read the next update record and return to the compare above. If not delete all, read and process each update record for this term. Output this inverted file record with the changes made, read the next inverted file record and return to the compare above.

5. End of File

   After either end of file, pad that file with high order values and process the remainder of the other file. After both end of files on the input tapes, close all three tape files, finish writing the index tables on disk and return to the supervisor program.

6. Process Update Record Routine

   There are three types of update records:

   a. add a document number to an inverted file record
   b. delete a document number from a record
   c. delete an entire document number string (main term)

The document numbers within each string are in numeric sequence. To add a document number to a string, find the location in the string where this document number fits numerically, expand the string and insert this document number. When a string becomes excessively long (perhaps 1000 document numbers per string), it is split into two separate logical records with the same main term. Each such string will be numbered sequentially and all but the last such string for each term will contain a continuation flag. To delete a document number from a string, locate the number to be deleted and shrink the string over that number. To delete a main term, all document number strings for that term are removed from the inverted file.

1. Output Routine

   If there are any changes to the inverted file record or any errors, one or more message notes are written on SYSLST. If the request is a delete with no errors exit this routine. If there was no change or if there were any errors, the unchanged input record is written on tape and disk, otherwise the new inverted file record is outputted. Return to the main program.

2. Output Disk Routine

   The disk output routine blocks the inverted file records and writes this data on the disk one full track at a time (3625 characters per track). As each track is completed and written on disk a track index is updated with the main term for the last record on that track. For each ten tracks a cylinder index is updated and for each 50 cylinders (10 tracks each cylinder) the master index is updated.

X'00'	a complete inverted file disk record is on this track.
X'01'	the first part of an inverted file disk record which is continued on the next track.
X'02'	the last part of an inverted file disk record which is continued from the prior track.
X'FF'	the communication vector on cylinder 1, track 0 of the inverted file disk. This contains the following: NAME DATE CREATED TRACK ADDRESSES OF CYL INDEX TABLES LAST TRACK WRITTEN TRACKS USED PER CYLINDER NUMBER OF DISK ERRORS NUMBER OF ALT. TRACKS USED MAX. LENGTH OF DISK RECORDS

X'22'	Master index on cylinder 1, track 0 of the inverted file disk.
X'82'	Cylinder index
X'43'	Track index

Input

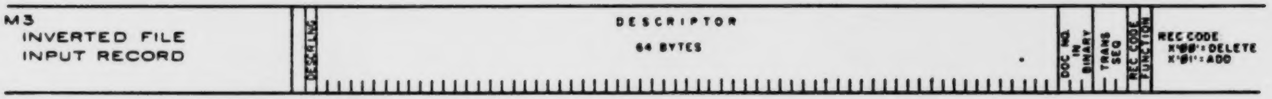
Two tapes serve as input to the inverted file update program.

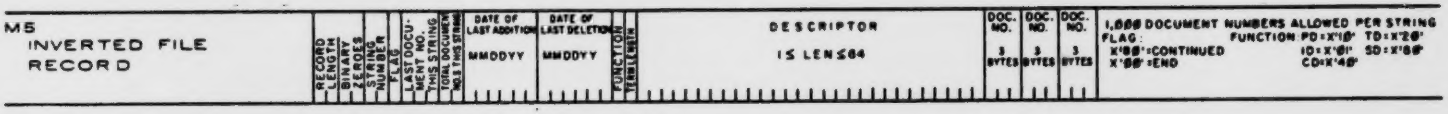
1. The prior inverted file tape is read in the format labelled M5 below. The records are variable in length and blocked.
2. The update created by MFMAINT and later sorted is in the format labelled M3 below.

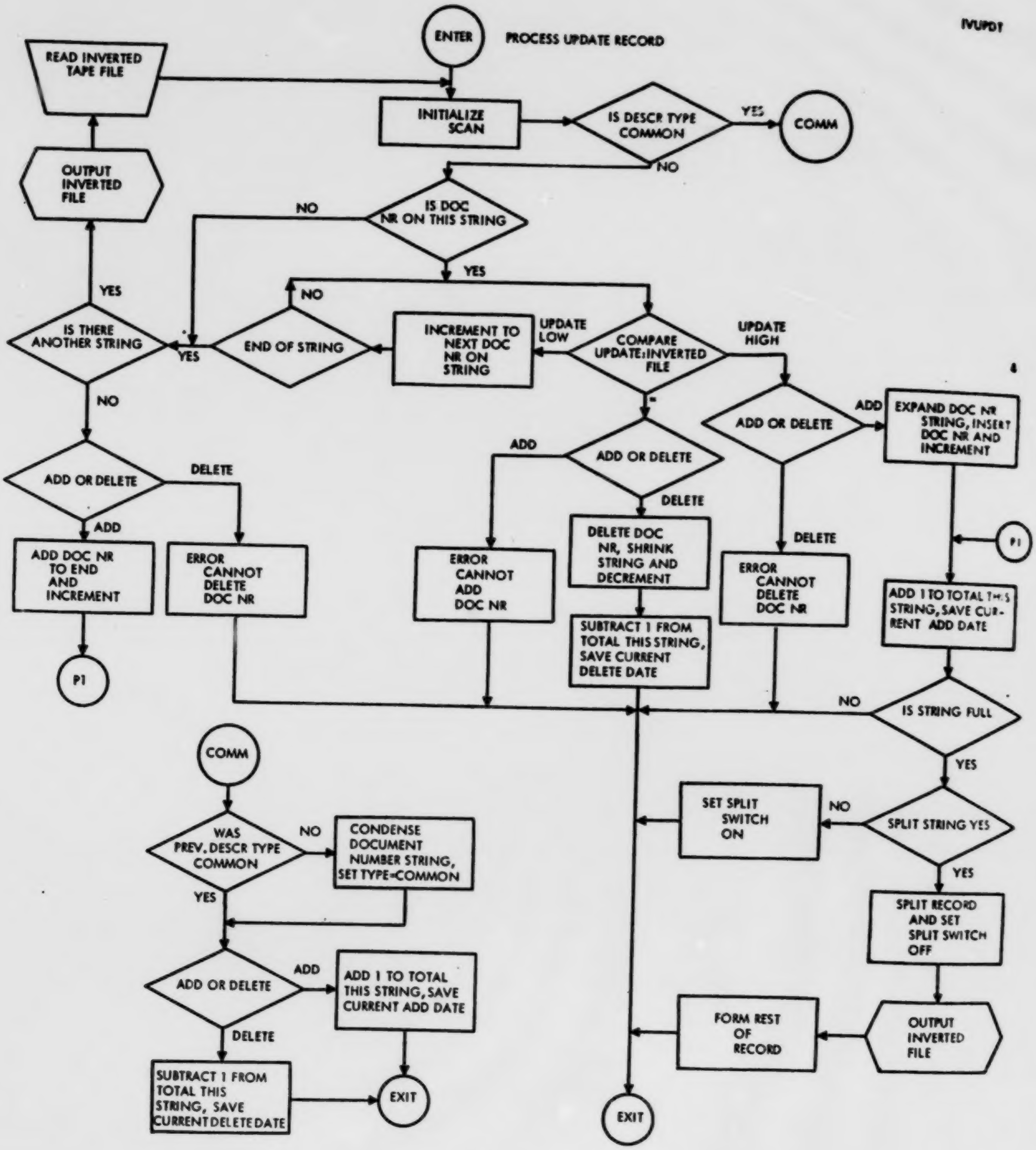
Output

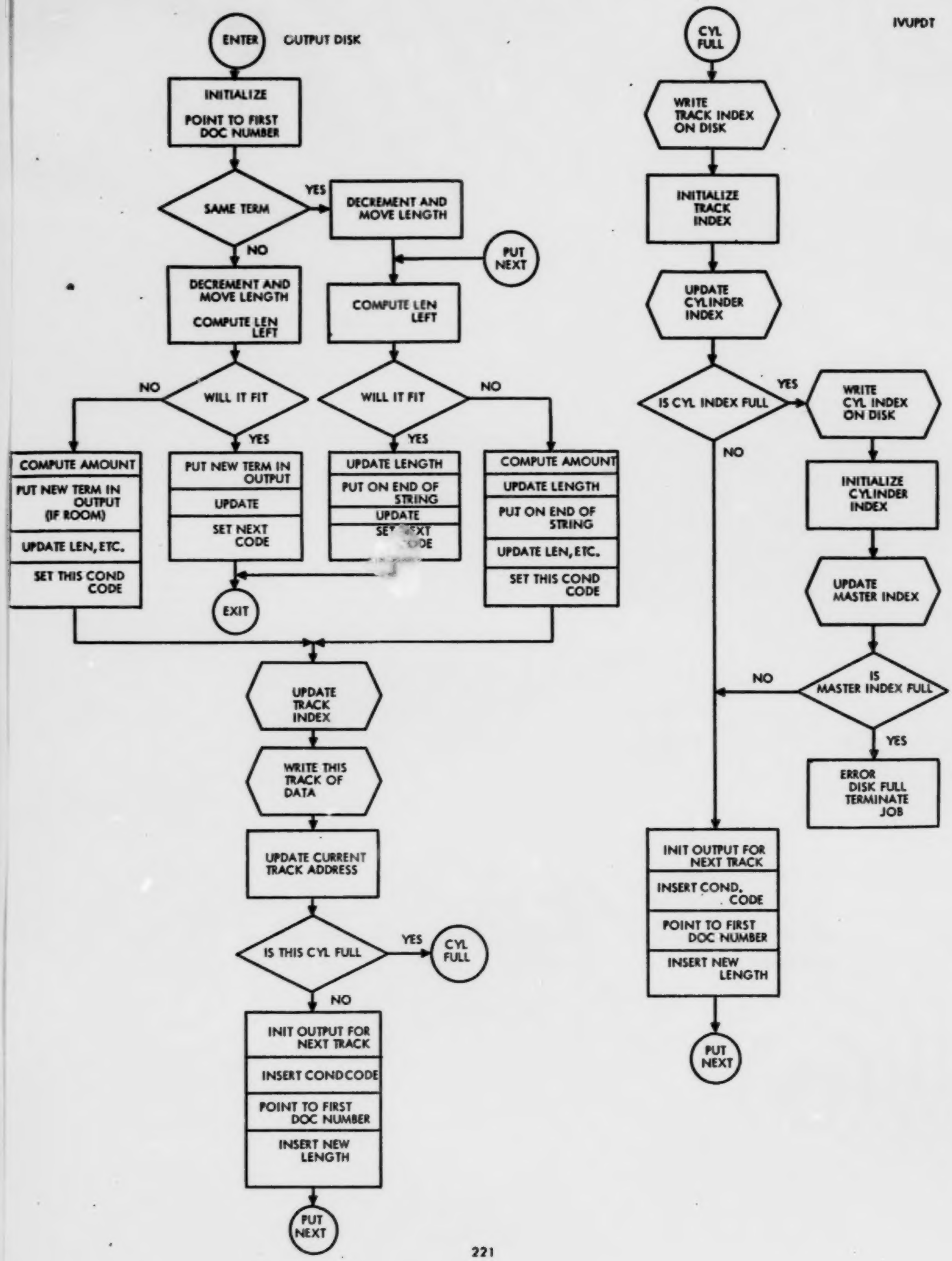
Tape and disk versions of the inverted file are created by the program. The tape format is shown below labelled M5. The disk format is the same except that the first 25 bytes are not included. For a more detailed description of the inverted file record, please refer to Appendix C.

INVERTED FILE UPDATE PROGRAM (IFUPDATE) RECORD FORMATS

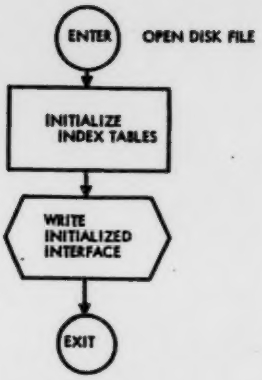

M3
INVERTED FILE
INPUT RECORD
DESCRLNG
DESCRIPTOR
64 BYTES
DOC NO. IN BINARY
TRANS SEQ
REC CODE
FUNCTION
REC CODE
X'00':DELETE
X'01':ADD

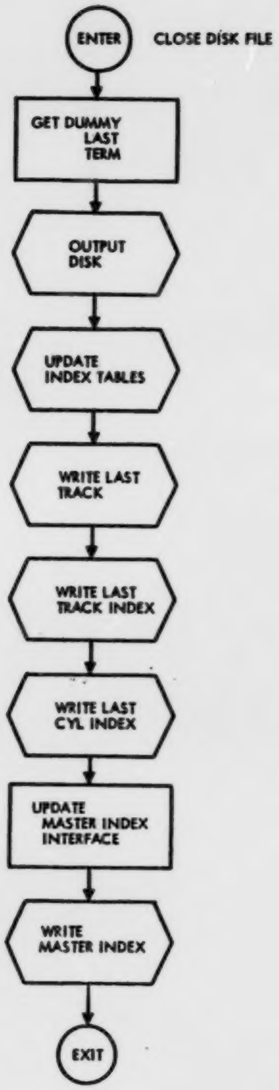

M5
INVERTED FILE
RECORD
RECORD LENGTH
BINARY ZEROES
STRING NUMBER
FLAG
LAST DOCU-MENT NO. THIS STRING
TOTAL DOCUMENT NO.S THIS STRING
DATE OF LAST ADDITION
MMDDYY
DATE OF LAST DELETION
MMDDYY
FUNCTION
TERMLENGTH
DESCRIPTOR
1 ≤ LEN ≤ 64
DOC. NO.
3 BYTES
DOC. NO.
3 BYTES
DOC. NO.
3 BYTES
1,000 DOCUMENT NUMBERS ALLOWED PER STRING
FLAG:
X'80':CONTINUED
X'00':END
FUNCTION: PD=X'10' TD=X'20'
ID=X'01' SD=X'80'
CD=X'40'


IVUPDT
ENTER
PROCESS UPDATE RECORD
READ INVERTED TAPE FILE
INITIALIZE SCAN
IS DESCR TYPE COMMON
YES
COMM
NO
OUTPUT INVERTED FILE
IS DOC NR ON THIS STRING
NO
YES
NO
IS THERE ANOTHER STRING
YES
END OF STRING
INCREMENT TO NEXT DOC NR ON STRING
UPDATE LOW
COMPARE UPDATE:INVERTED FILE
UPDATE HIGH
=
YES
NO
ADD OR DELETE
DELETE
ADD
ADD DOC NR TO END AND INCREMENT
ERROR CANNOT DELETE DOC NR
P1
ADD
ADD OR DELETE
DELETE
ERROR CANNOT ADD DOC NR
DELETE DOC NR, SHRINK STRING AND DECREMENT
SUBTRACT 1 FROM TOTAL THIS STRING, SAVE CURRENT DELETE DATE
ADD OR DELETE
ADD
EXPAND DOC NR STRING, INSERT DOC NR AND INCREMENT
DELETE
ERROR CANNOT DELETE DOC NR
P1
ADD 1 TO TOTAL THIS STRING, SAVE CURRENT ADD DATE
NO
IS STRING FULL
YES
SET SPLIT SWITCH ON
NO
SPLIT STRING YES
YES
SPLIT RECORD AND SET SPLIT SWITCH OFF
OUTPUT INVERTED FILE
FORM REST OF RECORD
EXIT
COMM
WAS PREV. DESCR TYPE COMMON
NO
CONDENSE DOCUMENT NUMBER STRING, SET TYPE=COMMON
YES
ADD OR DELETE
ADD
ADD 1 TO TOTAL THIS STRING, SAVE CURRENT ADD DATE
DELETE
SUBTRACT 1 FROM TOTAL THIS STRING, SAVE CURRENT DELETE DATE
EXIT


ENTER
OUTPUT DISK
INITIALIZE
POINT TO FIRST DOC NUMBER
SAME TERM
YES
NO
DECREMENT AND MOVE LENGTH
PUT NEXT
DECREMENT AND MOVE LENGTH
COMPUTE LEN LEFT
COMPUTE LEN LEFT
NO
WILL IT FIT
YES
WILL IT FIT
NO
YES
COMPUTE AMOUNT
PUT NEW TERM IN OUTPUT (IF ROOM)
UPDATE LEN, ETC.
SET THIS COND CODE
PUT NEW TERM IN OUTPUT
UPDATE
SET NEXT CODE
UPDATE LENGTH
PUT ON END OF STRING
UPDATE
SET NEXT CODE
COMPUTE AMOUNT
UPDATE LENGTH
PUT ON END OF STRING
UPDATE LEN, ETC.
SET THIS COND CODE
EXIT
UPDATE TRACK INDEX
WRITE THIS TRACK OF DATA
UPDATE CURRENT TRACK ADDRESS
IS THIS CYL FULL
YES
CYL FULL
NO
INIT OUTPUT FOR NEXT TRACK
INSERT COND CODE
POINT TO FIRST DOC NUMBER
INSERT NEW LENGTH
PUT NEXT
CYL FULL
WRITE TRACK INDEX ON DISK
INITIALIZE TRACK INDEX
UPDATE CYLINDER INDEX
IS CYL INDEX FULL
YES
WRITE CYL INDEX ON DISK
NO
INITIALIZE CYLINDER INDEX
UPDATE MASTER INDEX
NO
IS MASTER INDEX FULL
YES
ERROR DISK FULL TERMINATE JOB
INIT OUTPUT FOR NEXT TRACK
INSERT COND. CODE
POINT TO FIRST DOC NUMBER
INSERT NEW LENGTH
PUT NEXT

IVUPD7

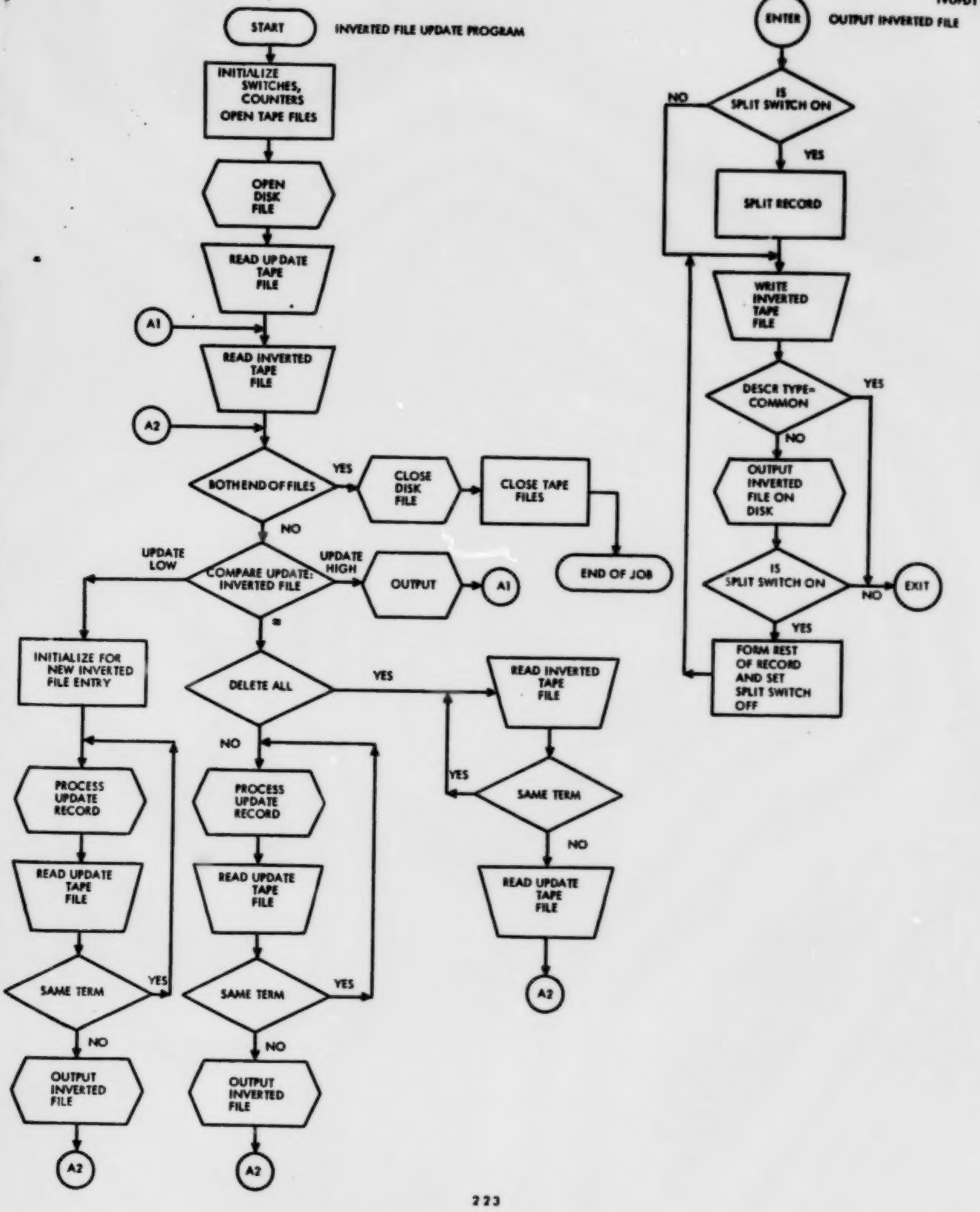

ENTER
OPEN DISK FILE
INITIALIZE INDEX TABLES
WRITE INITIALIZED INTERFACE
EXIT


ENTER
CLOSE DISK FILE
GET DUMMY LAST TERM
OUTPUT DISK
UPDATE INDEX TABLES
WRITE LAST TRACK
WRITE LAST TRACK INDEX
WRITE LAST CYL INDEX
UPDATE MASTER INDEX INTERFACE
WRITE MASTER INDEX
EXIT

IVUPDT

INVERTED FILE UPDATE PROGRAM

OUTPUT INVERTED FILE


START
INITIALIZE SWITCHES, COUNTERS
OPEN TAPE FILES
OPEN DISK FILE
READ UPDATE TAPE FILE
A1
READ INVERTED TAPE FILE
A2
BOTH END OF FILES
YES
CLOSE DISK FILE
CLOSE TAPE FILES
END OF JOB
NO
UPDATE LOW
COMPARE UPDATE: INVERTED FILE
UPDATE HIGH
OUTPUT
A1
=
INITIALIZE FOR NEW INVERTED FILE ENTRY
PROCESS UPDATE RECORD
READ UPDATE TAPE FILE
SAME TERM
YES
NO
OUTPUT INVERTED FILE
A2
DELETE ALL
YES
NO
PROCESS UPDATE RECORD
READ UPDATE TAPE FILE
SAME TERM
YES
NO
OUTPUT INVERTED FILE
A2
READ INVERTED TAPE FILE
SAME TERM
YES
NO
READ UPDATE TAPE FILE
A2
ENTER
IS SPLIT SWITCH ON
NO
YES
SPLIT RECORD
WRITE INVERTED TAPE FILE
DESCR TYPE= COMMON
YES
NO
OUTPUT INVERTED FILE ON DISK
IS SPLIT SWITCH ON
NO
EXIT
YES
FORM REST OF RECORD AND SET SPLIT SWITCH OFF

INVERTED FILE PRINT PROGRAM (IFPRNT)

Processing

The program opens the tape file and reads the first record. The data is reformatted and printed as soon as a line is full (i.e., when the next document number will not fit or when a new descriptor is encountered). Common descriptors (which are included in the tape version of the inverted file) are printed with a message indicating that they are, in fact, common descriptors. Since no document numbers are included with common descriptors, only the descriptor and the message appear. Special descriptors which were input to MFEDIT preceded by a dollar sign ($) are printed last. As soon as an end-of-file is encountered on the input tape, the file is closed and processing terminates.

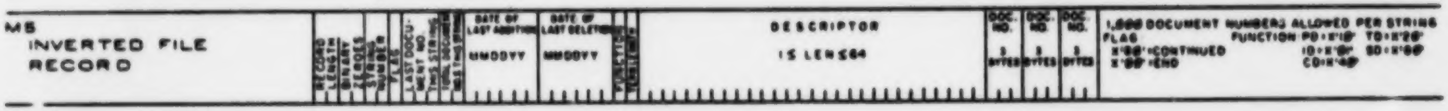
Input

The tape version of the inverted file (format M5) is input to IFPRNT.

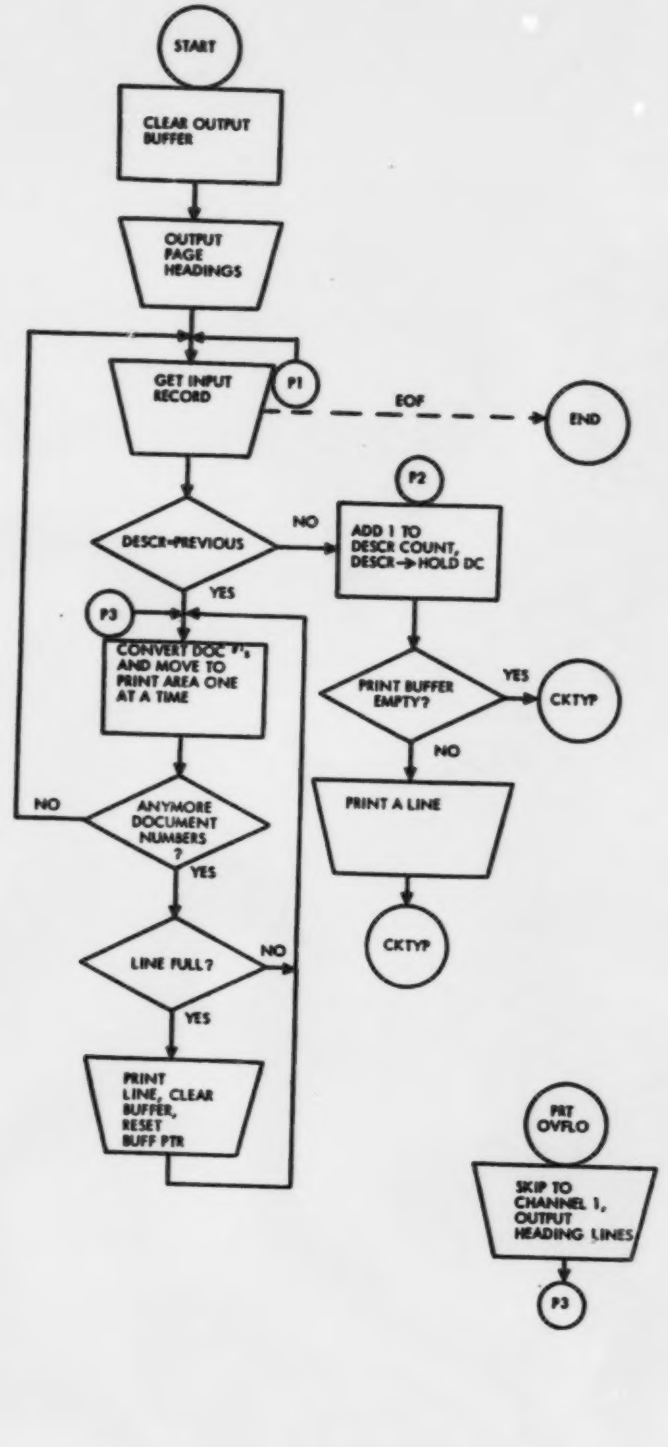
Output

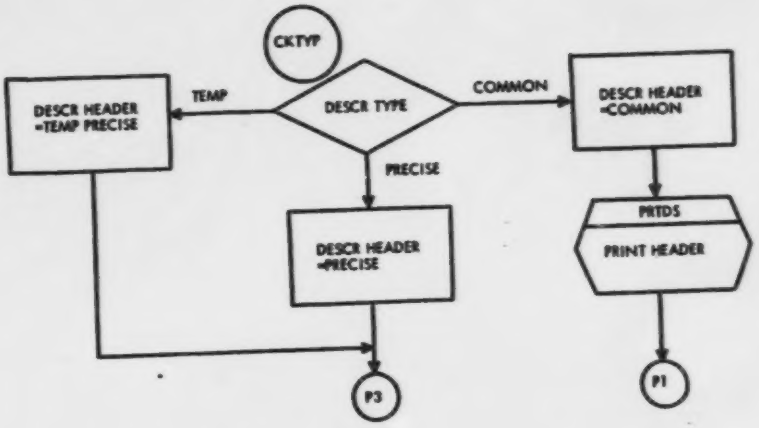
A printed listing of the entire file is produced on SYSLST.

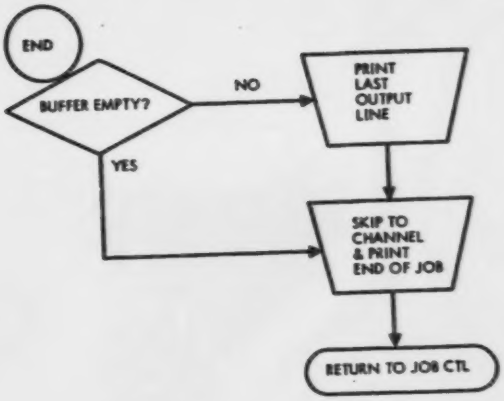
INVERTED FILE PRINT PROGRAM (IFPRNT) RECORD FORMAT


M5
INVERTED FILE
RECORD
RECORD LENGTH
BINARY ZEROES
STRING NUMBER
FLAG
LAST DOCUMENT NO. THIS STRING
TOTAL DOCUMENTS THIS STRING
DATE OF LAST ADDITION
MMDDYY
DATE OF LAST DELETION
MMDDYY
FUNCTION
DESCRIPTOR
1 ≤ LEN ≤ 64
DOC. NO.
3 BYTES
DOC. NO.
3 BYTES
DOC. NO.
3 BYTES
1,000 DOCUMENT NUMBERS ALLOWED PER STRING
FLAG
X'00'=CONTINUED
X'FF'=END
FUNCTION PD=X'10' TO=X'20'
ID=X'08' SD=X'80'
CD=X'40'

IFPRNT


START
CLEAR OUTPUT BUFFER
OUTPUT PAGE HEADINGS
GET INPUT RECORD
P1
EOF
END
DESCR=PREVIOUS
NO
P2
ADD 1 TO DESCR COUNT, DESCR→HOLD DC
YES
P3
CONVERT DOC #'s AND MOVE TO PRINT AREA ONE AT A TIME
PRINT BUFFER EMPTY?
YES
CKTYP
NO
NO
ANYMORE DOCUMENT NUMBERS ?
PRINT A LINE
YES
CKTYP
LINE FULL?
NO
YES
PRINT LINE, CLEAR BUFFER, RESET BUFF PTR
PRT OVFLO
SKIP TO CHANNEL 1, OUTPUT HEADING LINES
P3


CKTYP
DESCR TYPE
TEMP
COMMON
PRECISE
DESCR HEADER =TEMP PRECISE
DESCR HEADER =PRECISE
DESCR HEADER =COMMON
PRTDS
PRINT HEADER
P3
P1


END
BUFFER EMPTY?
NO
YES
PRINT LAST OUTPUT LINE
SKIP TO CHANNEL & PRINT END OF JOB
RETURN TO JOB CTL

INVERTED FILE STATISTICS PRINT PROGRAM (IVPRINT)

Processing

The tape and card input file are opened and the title card is read into the title area. The records from the inverted file tape are read and the statistical data contained in the first 25 bytes plus the descriptor are reformatted and moved to an output buffer. As soon as the buffer is filled, the printer channel is queried to determine whether or not it is busy. If busy, IVPRINT waits until it is free. Then the buffer just filled is printed and processing continues. Since records may be continued from one physical block to the next, no data is moved until the most current statistical data for the term is compiled. An end-of-file on the input tape terminates processing.

Input

The tape version of the inverted file is in the format labelled M5 below. For a more detailed description of the inverted file record, please refer to Appendix C.

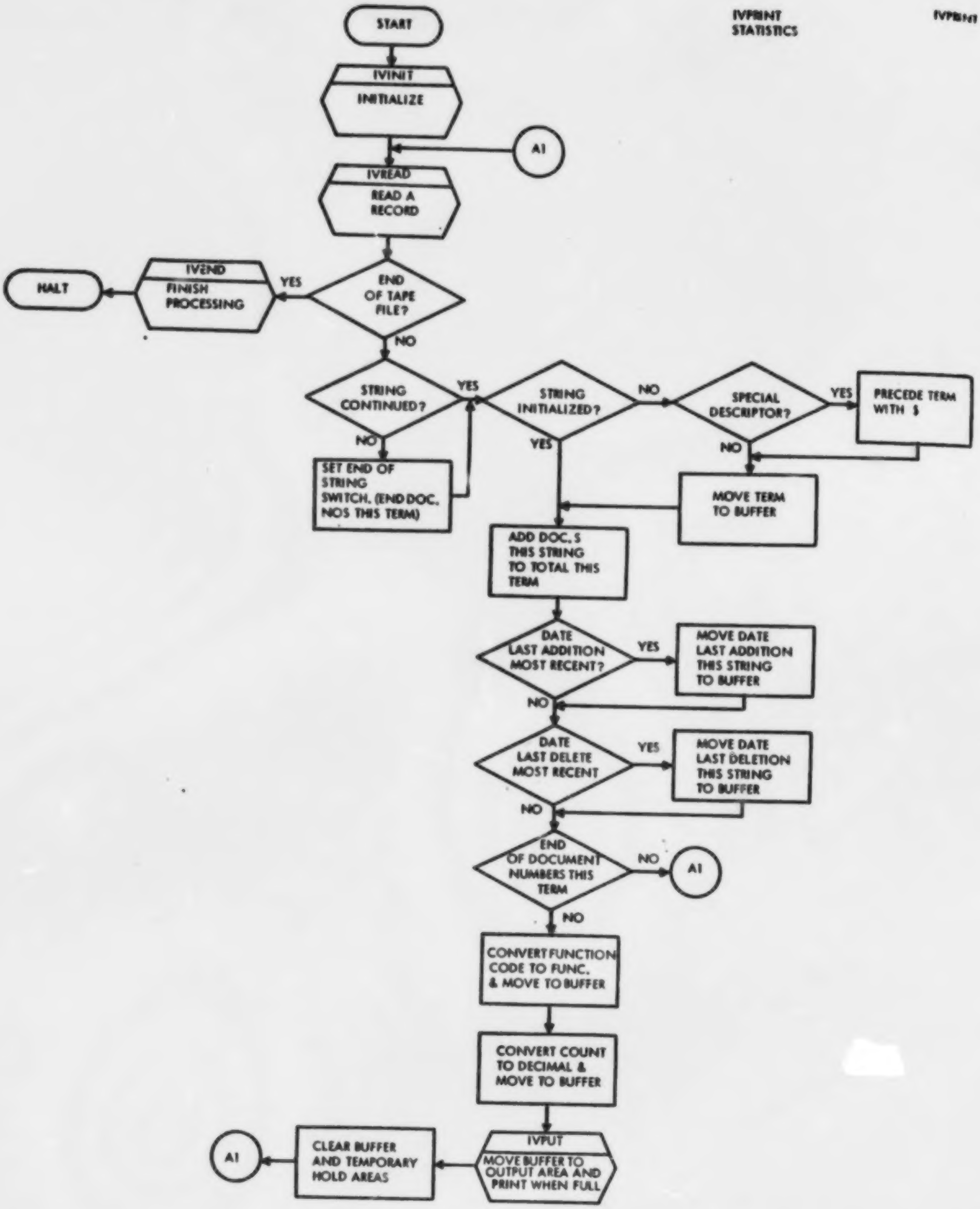
Output

The statistical data for the inverted file is printed on SYSLST.

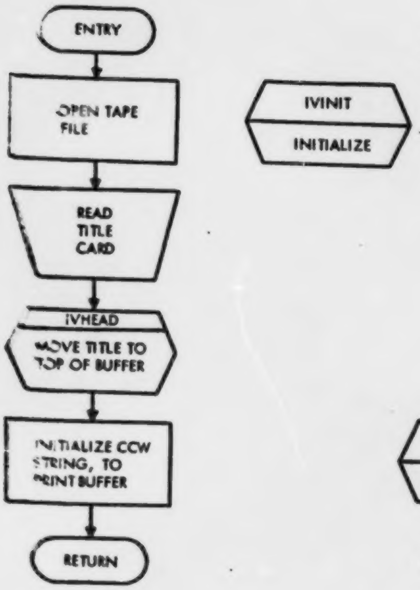
INVERTED FILE STATISTICS PRINT PROGRAM (IVPRINT) RECORD FORMAT

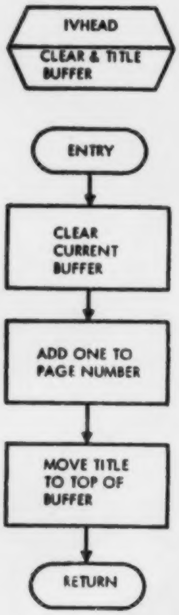
M5 INVERTED FILE RECORD	RECORD LENGTH	BINARY ZEROES	STRING NUMBER	FLAG	LAST DOCUMENT NO. THIS STRING	TOTAL DOCUMENT NOS THIS STRING	DATE OF LAST ADDITION	DATE OF LAST DELETION	FUNCTION	TERM LENGTH	DESCRIPTOR	DOC. NO.	DOC. NO.	DOC. NO.	1,000 DOCUMENT NUMBERS ALLOWED PER STRING
							MMDDYY	MMDDYY			1 ≤ LEN ≤ 64	3 BYTES	3 BYTES	3 BYTES	FLAG X'00' : CONTINUED X'80' : END FUNCTION: PD=X'10' TD=X'20' ID=X'01' SD=X'80' CD=X'40'

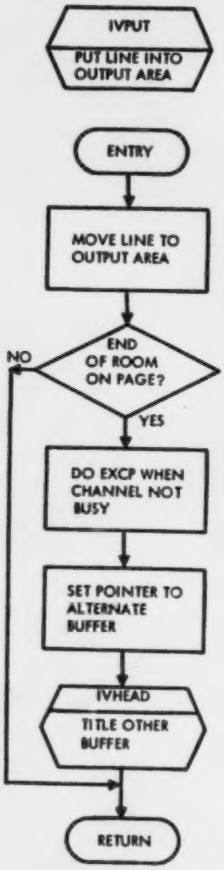
IVPRINT
STATISTICS

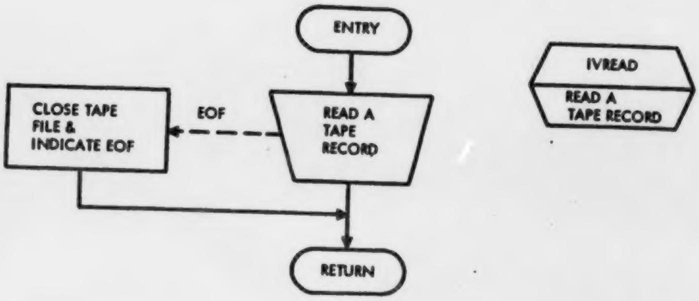

START
IVINIT
INITIALIZE
A1
IVREAD
READ A RECORD
END OF TAPE FILE?
YES
IVEND
FINISH PROCESSING
HALT
NO
STRING CONTINUED?
NO
SET END OF STRING SWITCH, (END DOC. NOS THIS TERM)
YES
STRING INITIALIZED?
NO
SPECIAL DESCRIPTOR?
YES
PRECEDE TERM WITH $
NO
MOVE TERM TO BUFFER
YES
ADD DOC. S THIS STRING TO TOTAL THIS TERM
DATE LAST ADDITION MOST RECENT?
YES
MOVE DATE LAST ADDITION THIS STRING TO BUFFER
NO
DATE LAST DELETE MOST RECENT
YES
MOVE DATE LAST DELETION THIS STRING TO BUFFER
NO
END OF DOCUMENT NUMBERS THIS TERM
NO
A1
NO
CONVERT FUNCTION CODE TO FUNC. & MOVE TO BUFFER
CONVERT COUNT TO DECIMAL & MOVE TO BUFFER
IVPUT
MOVE BUFFER TO OUTPUT AREA AND PRINT WHEN FULL
CLEAR BUFFER AND TEMPORARY HOLD AREAS
A1

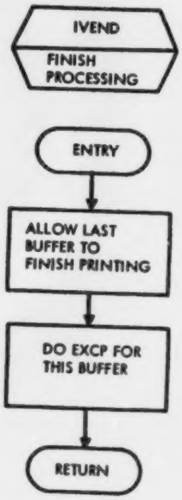
IVPRINT
STATISTICS


IVINIT
INITIALIZE
ENTRY
OPEN TAPE FILE
READ TITLE CARD
IVHEAD
MOVE TITLE TO TOP OF BUFFER
INITIALIZE CCW STRING, TO PRINT BUFFER
RETURN


IVHEAD
CLEAR & TITLE BUFFER
ENTRY
CLEAR CURRENT BUFFER
ADD ONE TO PAGE NUMBER
MOVE TITLE TO TOP OF BUFFER
RETURN


IVPUT
PUT LINE INTO OUTPUT AREA
ENTRY
MOVE LINE TO OUTPUT AREA
END OF ROOM ON PAGE?
NO
YES
DO EXCP WHEN CHANNEL NOT BUSY
SET POINTER TO ALTERNATE BUFFER
IVHEAD
TITLE OTHER BUFFER
RETURN


IVREAD
READ A TAPE RECORD
ENTRY
READ A TAPE RECORD
EOF
CLOSE TAPE FILE & INDICATE EOF
RETURN


IVEND
FINISH PROCESSING
ENTRY
ALLOW LAST BUFFER TO FINISH PRINTING
DO EXCP FOR THIS BUFFER
RETURN

UPDATE CROSS REFERENCE FILE (UPDXREF) PROCESSING LOGIC

Main Routine

The main routine of UPDXREF handles the reading of tapes and the comparisons between numbers of each record. When an end-of-file is encountered on either tape, the input area reserved for the external number of that file is filled with maximum values so that a compare will always indicate that the external number on the other tape is low. In addition, the main routine supervises the moving of the records to an output area and the updating of the counter which keeps track of the number of records in the output area. As soon as that area is full, the OUTPUT routine is entered.

Output Routine

This routine handles the writing of the data on tape and disk. In addition, it updates the index for the disk by placing the last external number for the current track and the track identification in the index area.

End-of-File Routine

When both tapes have reached the end-of-file, this routine is entered. It finishes writing the remaining data on tape and disk. Then the index is written on the disk, the disk overflow area is cleared, and processing terminates.

Input

Two tapes serve as input to UPDXREF.

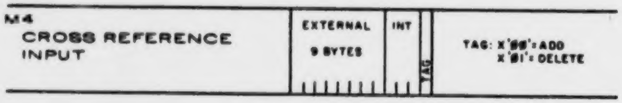
1. The tape created by MFMAINT is sorted into ascending external number sequence and provides update input to the program. Its format is shown below labelled M4.

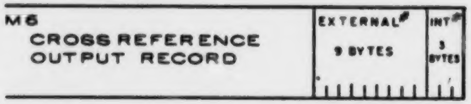
2. The prior version of the cross reference file is read in the format labelled M6 below. The records are fixed length and blocked.

Output

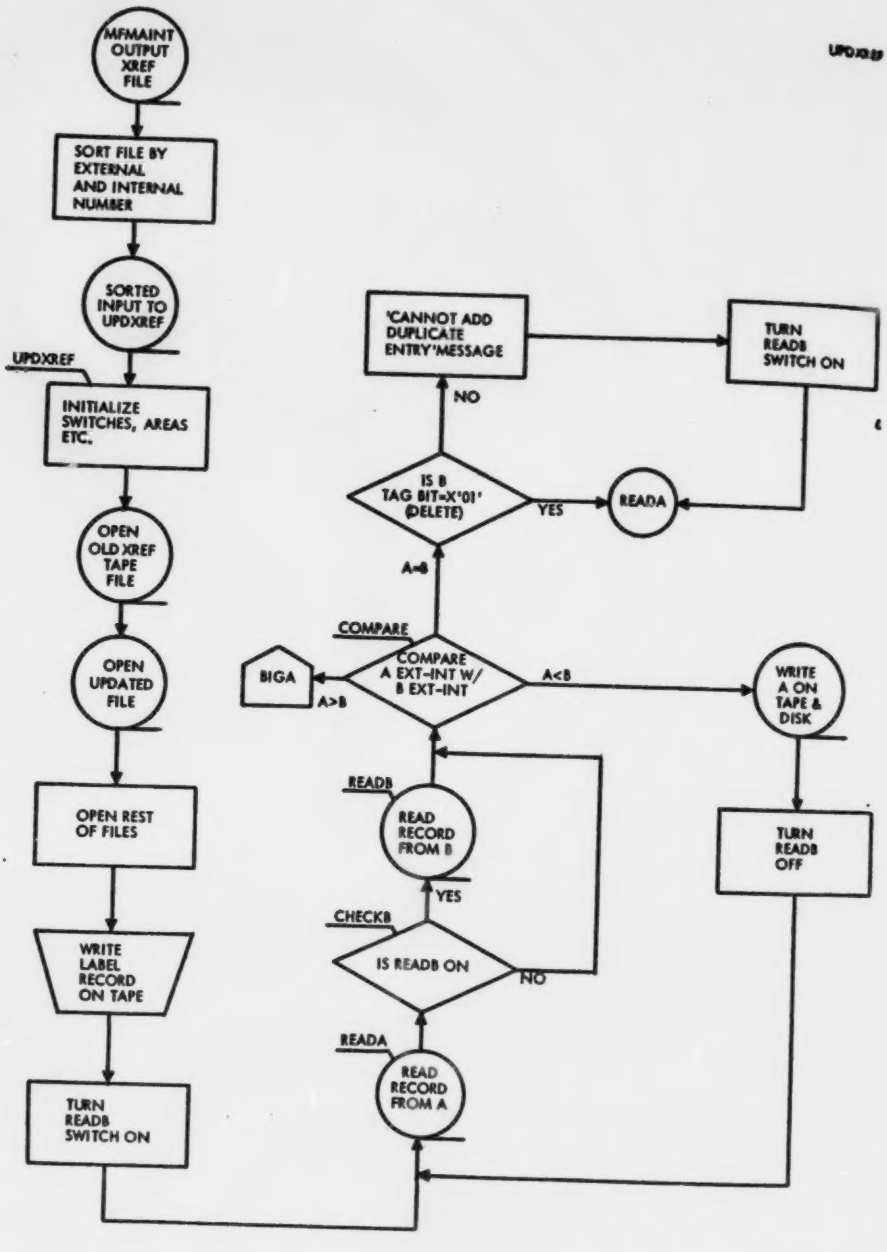
Both a tape and a direct access version of the cross reference file are created. The two are in exactly the same format, labelled M6 below.

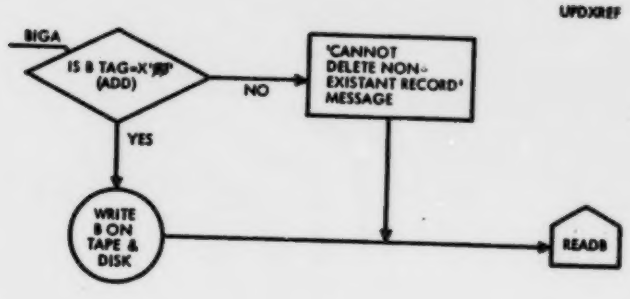
UPDATE CROSS REFERENCE FILE (UPDXREF) RECORD FORMATS

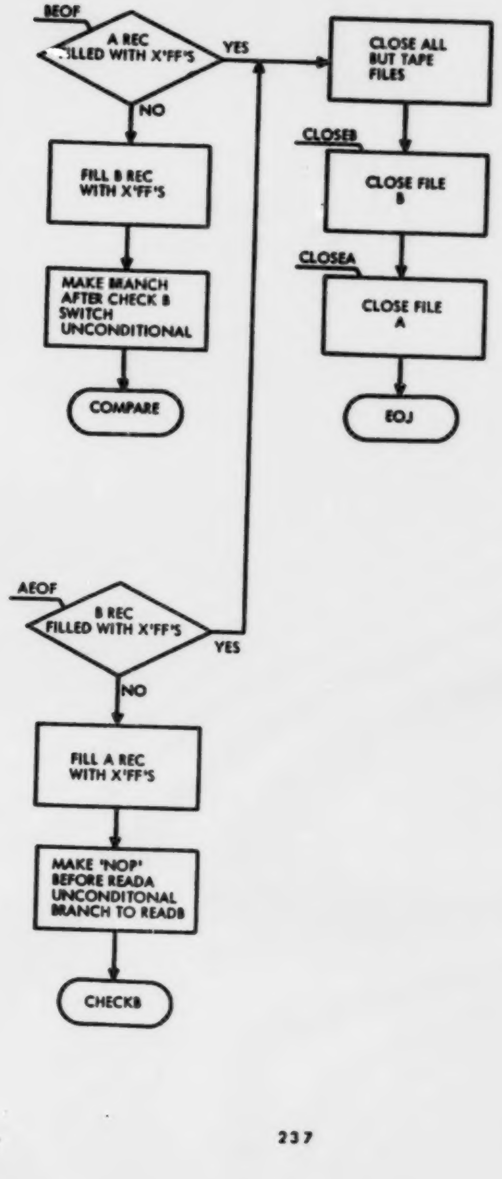

M4
CROSS REFERENCE
INPUT
EXTERNAL
9 BYTES
INT
TAG
TAG: X'00': ADD
X'01': DELETE


M6
CROSS REFERENCE
OUTPUT RECORD
EXTERNAL
9 BYTES
INT
3
BYTES

UPDXREF


MFMAINT OUTPUT XREF FILE
SORT FILE BY EXTERNAL AND INTERNAL NUMBER
SORTED INPUT TO UPDXREF
UPDXREF
INITIALIZE SWITCHES, AREAS ETC.
OPEN OLD XREF TAPE FILE
OPEN UPDATED FILE
OPEN REST OF FILES
WRITE LABEL RECORD ON TAPE
TURN READB SWITCH ON
READA
READ RECORD FROM A
CHECKB
IS READB ON
YES
NO
READB
READ RECORD FROM B
COMPARE
COMPARE A EXT-INT W/ B EXT-INT
A>B
BIGA
A<B
WRITE A ON TAPE & DISK
TURN READB OFF
A=B
IS B TAG BIT=X'01' (DELETE)
YES
READA
NO
'CANNOT ADD DUPLICATE ENTRY'MESSAGE
TURN READB SWITCH ON


BIGA
IS B TAG=X'[illegible]' (ADD)
NO
'CANNOT DELETE NON-EXISTANT RECORD' MESSAGE
YES
WRITE B ON TAPE & DISK
READB


BEOF
A REC FILLED WITH X'FF'S
YES
CLOSE ALL BUT TAPE FILES
NO
FILL B REC WITH X'FF'S
MAKE BRANCH AFTER CHECK B SWITCH UNCONDITIONAL
COMPARE
CLOSEB
CLOSE FILE B
CLOSEA
CLOSE FILE A
EOJ
AEOF
B REC FILLED WITH X'FF'S
YES
NO
FILL A REC WITH X'FF'S
MAKE 'NOP' BEFORE READA UNCONDITONAL BRANCH TO READB
CHECKB

CROSS REFERENCE PRINT PROGRAM (XREFPRNT)

Processing

The files are opened and the first record from the cross reference tape is read. The internal number is converted to decimal and is placed in the output buffer below the external number. The next record is read. If the external numbers are equal (i.e., if the cross reference override feature was exercised in MFEDIT) only the internal number will appear. Otherwise, both members of the external/internal pair will be placed in the buffer.

Columns in the buffer are filled vertically. A maximum of eight columns will appear on a page. As soon as the buffer is full and the printer is available, an EXCP is issued and the printing of the buffer begins. Then processing resumes as above. An end-of-file condition on the input tape terminates processing.

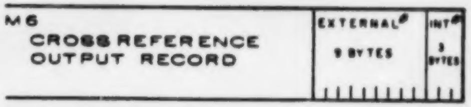
Input

The tape version of the cross reference file (labelled M6 below) is input to XREFPRNT. The records are fixed length and blocked.

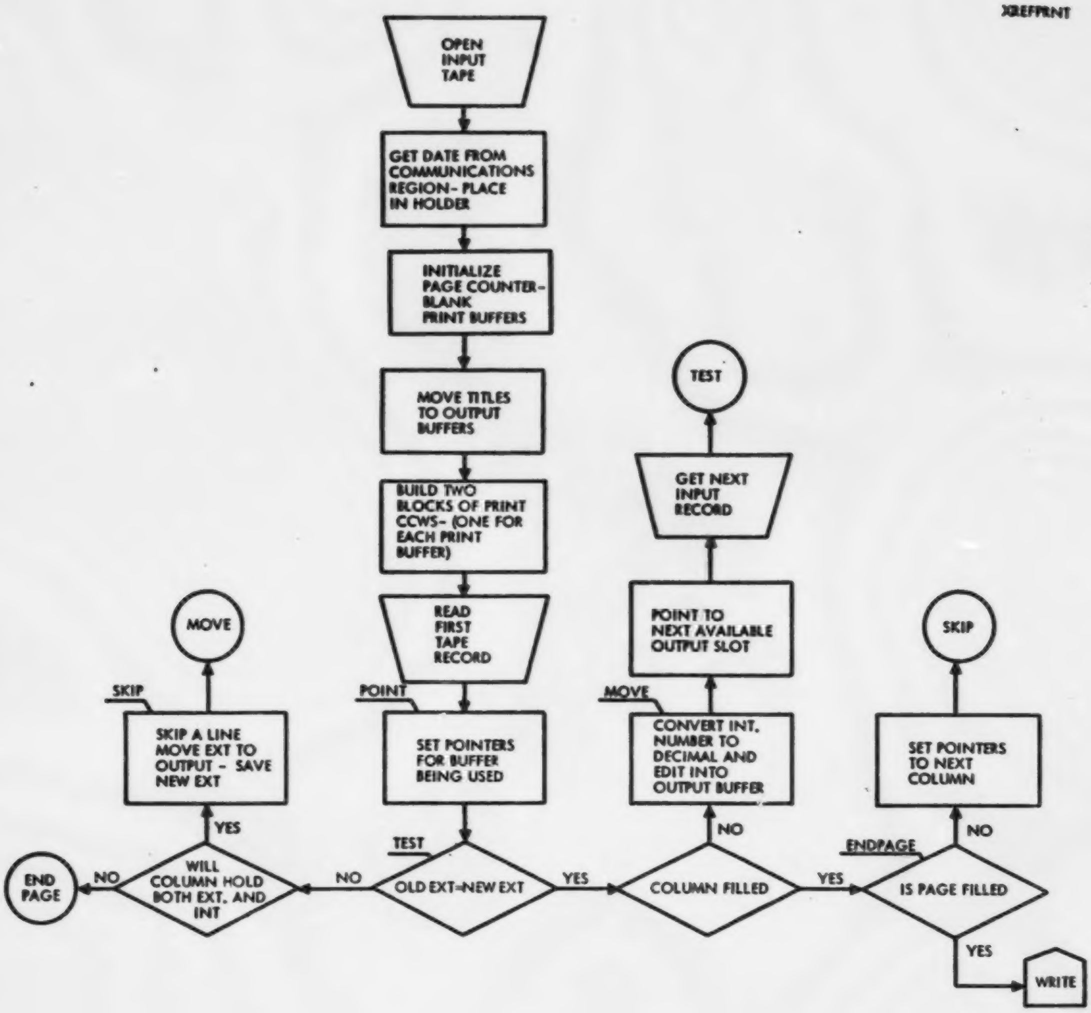
Output

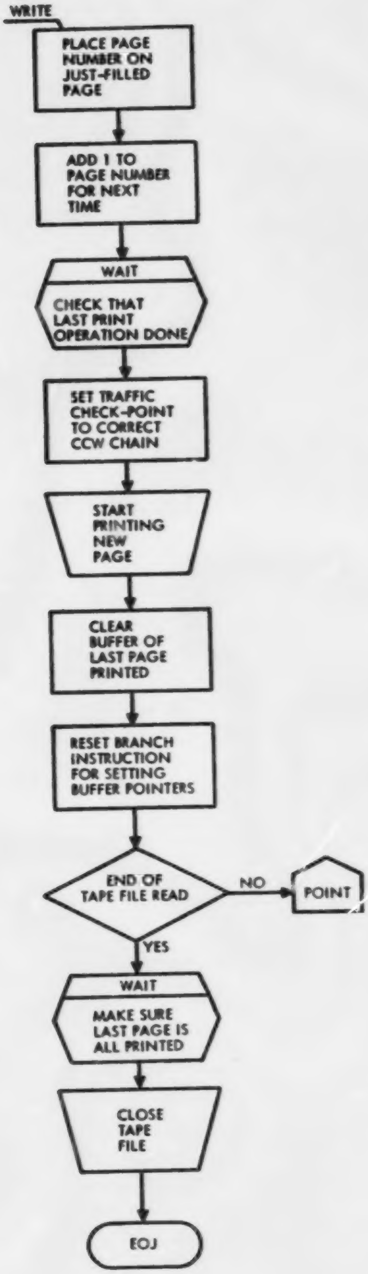
The printed cross reference file is produced on SYSLST.

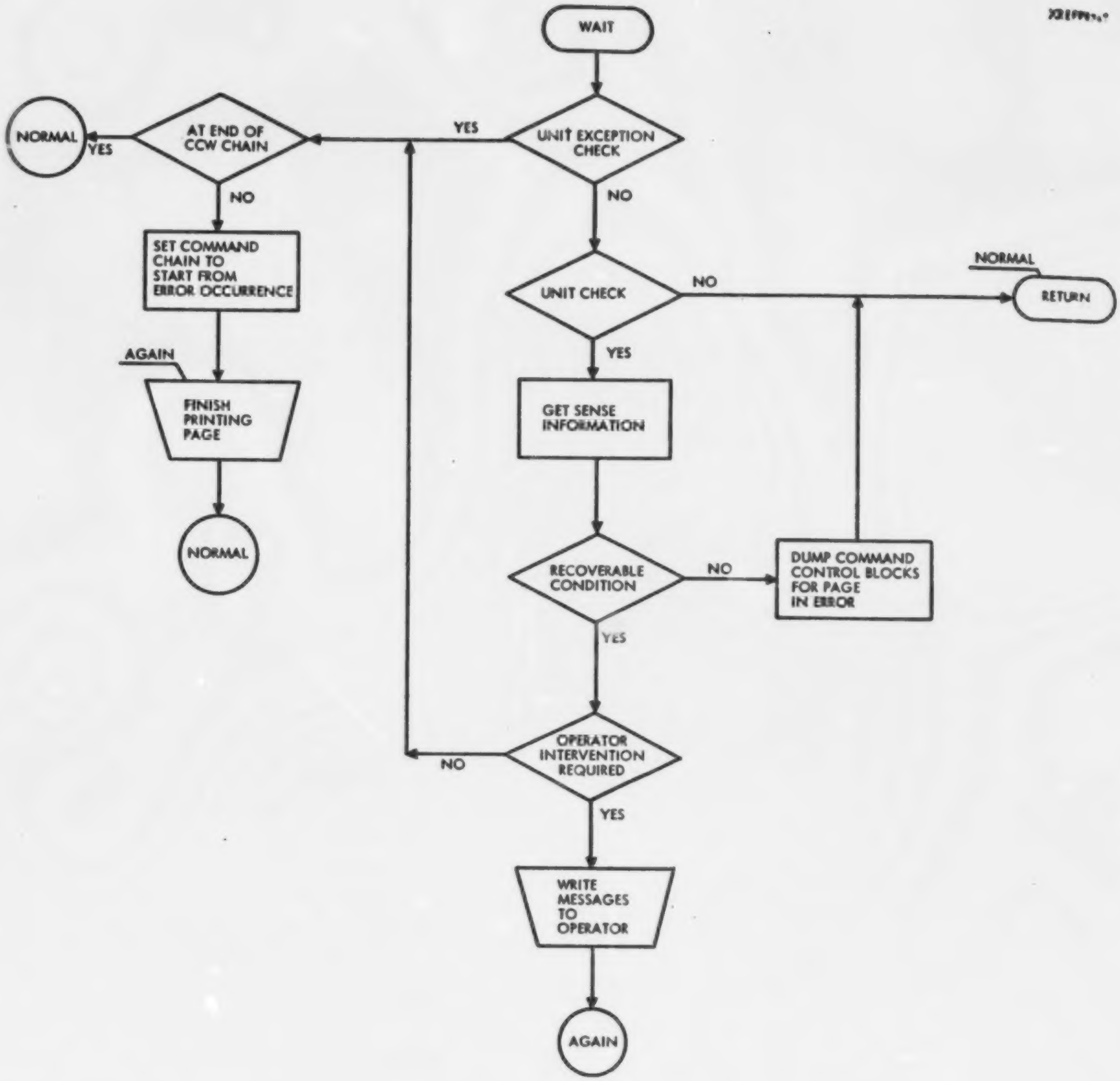
CROSS REFERENCE PRINT PROGRAM (XREFPRNT) RECORD FORMAT


M6
CROSS REFERENCE
OUTPUT RECORD
EXTERNAL
9 BYTES
INT
3
BYTES

XREFPRNT


OPEN INPUT TAPE
GET DATE FROM COMMUNICATIONS REGION- PLACE IN HOLDER
INITIALIZE PAGE COUNTER- BLANK PRINT BUFFERS
MOVE TITLES TO OUTPUT BUFFERS
BUILD TWO BLOCKS OF PRINT CCWS- (ONE FOR EACH PRINT BUFFER)
READ FIRST TAPE RECORD
POINT
SET POINTERS FOR BUFFER BEING USED
TEST
OLD EXT=NEW EXT
NO
YES
WILL COLUMN HOLD BOTH EXT. AND INT
NO
END PAGE
YES
SKIP
SKIP A LINE MOVE EXT TO OUTPUT - SAVE NEW EXT
MOVE
COLUMN FILLED
NO
MOVE
CONVERT INT. NUMBER TO DECIMAL AND EDIT INTO OUTPUT BUFFER
POINT TO NEXT AVAILABLE OUTPUT SLOT
GET NEXT INPUT RECORD
TEST
YES
ENDPAGE
IS PAGE FILLED
NO
SET POINTERS TO NEXT COLUMN
SKIP
YES
WRITE


WRITE
PLACE PAGE NUMBER ON JUST-FILLED PAGE
ADD 1 TO PAGE NUMBER FOR NEXT TIME
WAIT
CHECK THAT LAST PRINT OPERATION DONE
SET TRAFFIC CHECK-POINT TO CORRECT CCW CHAIN
START PRINTING NEW PAGE
CLEAR BUFFER OF LAST PAGE PRINTED
RESET BRANCH INSTRUCTION FOR SETTING BUFFER POINTERS
END OF TAPE FILE READ
NO
POINT
YES
WAIT
MAKE SURE LAST PAGE IS ALL PRINTED
CLOSE TAPE FILE
EOJ


WAIT
NORMAL
YES
AT END OF CCW CHAIN
YES
UNIT EXCEPTION CHECK
NO
NO
SET COMMAND CHAIN TO START FROM ERROR OCCURRENCE
NORMAL
UNIT CHECK
NO
RETURN
YES
AGAIN
FINISH PRINTING PAGE
GET SENSE INFORMATION
NORMAL
RECOVERABLE CONDITION
NO
DUMP COMMAND CONTROL BLOCKS FOR PAGE IN ERROR
YES
NO
OPERATOR INTERVENTION REQUIRED
YES
WRITE MESSAGES TO OPERATOR
AGAIN

ID-FILE UPDATE PROGRAM (IDUPDATE) PROCESSING LOGIC

IDUPDATE reads records from revisions tape and merges them with the records from the prior ID-file. When the document numbers are equal and a delete tape is set in the revision record, the prior ID-file record is written onto a print tape. This can be printed using MPRINT2. Otherwise the input record is written on the output tape. When an end of file is read from the revisions tape, records are read from the additions tape and written at the end of the new ID-file tape until an end of file is read from the additions tape, which terminates the program.

Input

Three tapes may provide input to IDUPDATE.

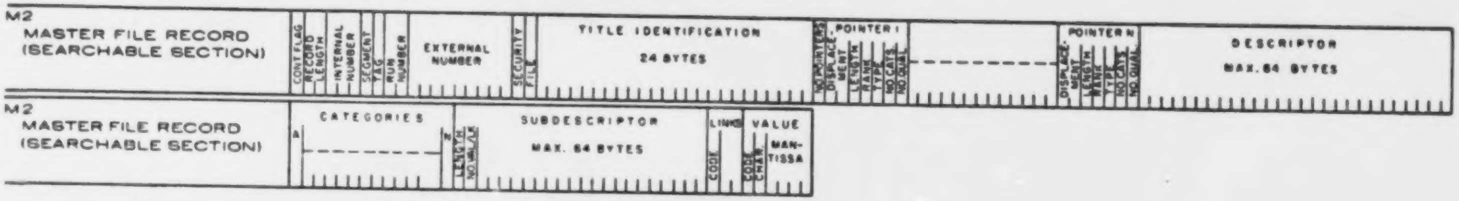
1. The prior ID. file
2. Change data from MFMAINT
3. Alterations to the file accomplished by running EDITBC (block change edit) and DUMPCELL.

These three tapes are in the format shown below labelled M2. It is identical to the master file record format except that no free text segments are included.

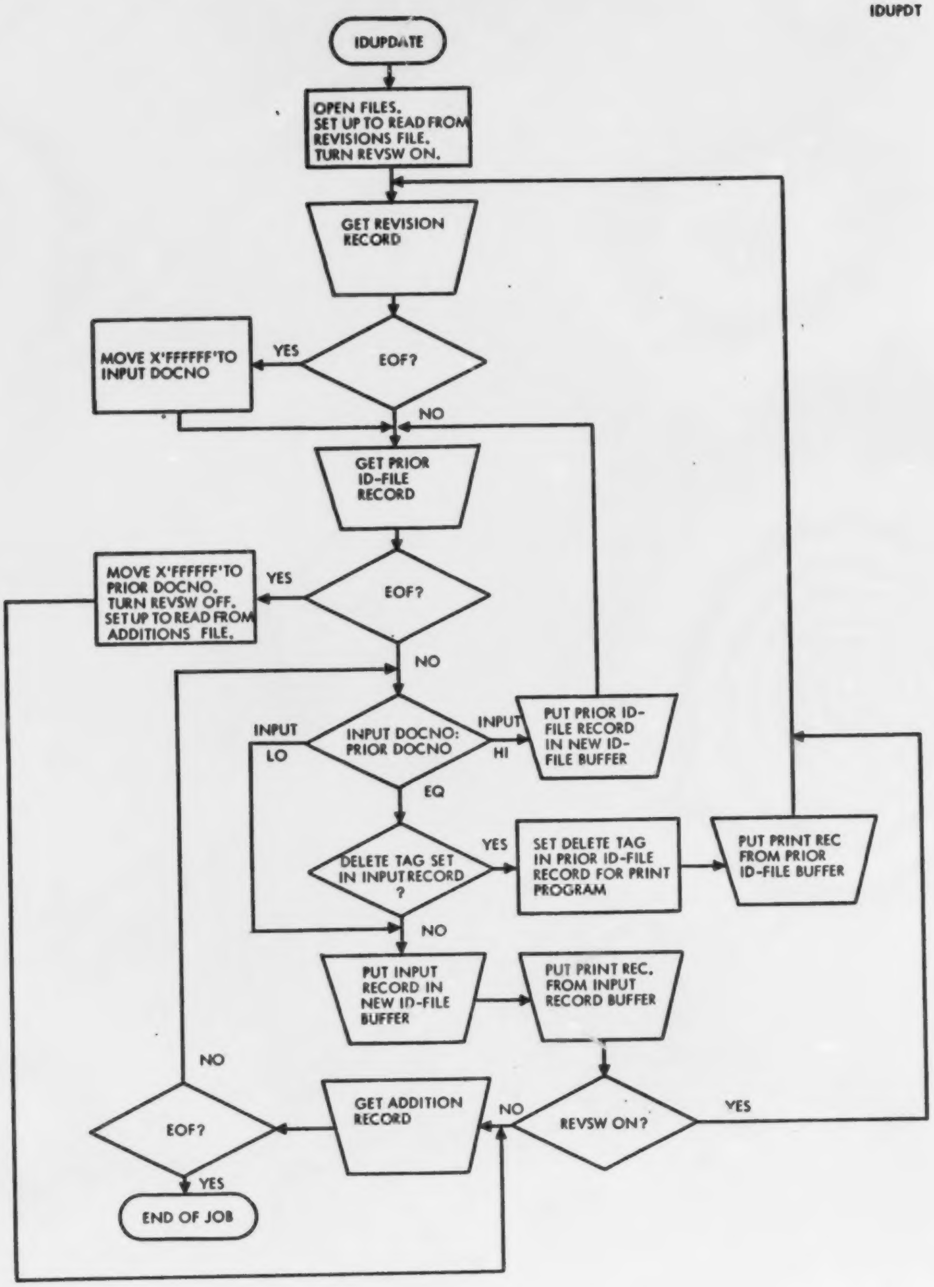
Output

The ID. file is written on magnetic tape in the format (M2) shown below.

ID. FILE UPDATE PROGRAM (IDUPDATE) RECORD FORMAT


M2
MASTER FILE RECORD (SEARCHABLE SECTION)
CONT FLAG
RECORD LENGTH
INTERNAL NUMBER
SEGMENT
TAG
RUN NUMBER
EXTERNAL NUMBER
SECURITY
FILE
TITLE IDENTIFICATION
24 BYTES
NO POINTERS
POINTER 1
DISPLACE-MENT
LENGTH
RANK
TYPE
NO CATS
NO QUAL
POINTER N
DISPLACE-MENT
LENGTH
RANK
TYPE
NO CATS
NO QUAL
DESCRIPTOR
MAX. 64 BYTES
M2
MASTER FILE RECORD (SEARCHABLE SECTION)
CATEGORIES
A
N
LENGTH
NO VALUE
SUBDESCRIPTOR
MAX. 64 BYTES
LINK
CODE
VALUE
CODE
CHAR.
MAN-TISSA

IDUPDT



IDUPDATE
OPEN FILES. SET UP TO READ FROM REVISIONS FILE. TURN REVSW ON.
GET REVISION RECORD
EOF?
YES
MOVE X'FFFFFF'TO INPUT DOCNO
NO
GET PRIOR ID-FILE RECORD
EOF?
YES
MOVE X'FFFFFF'TO PRIOR DOCNO. TURN REVSW OFF. SET UP TO READ FROM ADDITIONS FILE.
NO
INPUT DOCNO: PRIOR DOCNO
INPUT LO
INPUT HI
PUT PRIOR ID-FILE RECORD IN NEW ID-FILE BUFFER
EQ
DELETE TAG SET IN INPUT RECORD ?
YES
SET DELETE TAG IN PRIOR ID-FILE RECORD FOR PRINT PROGRAM
PUT PRINT REC FROM PRIOR ID-FILE BUFFER
NO
PUT INPUT RECORD IN NEW ID-FILE BUFFER
PUT PRINT REC. FROM INPUT RECORD BUFFER
REVSW ON?
YES
NO
GET ADDITION RECORD
EOF?
NO
YES
END OF JOB

MASTER FILE PRINT PROGRAM (MPRINT1) PROCESSING LOGIC

1. Main Routine

The main routine handles the reading of tape currently being processed, and the reformatting of the data on that tape. On the first pass the record type is tested. If the record is an Addition it is processed. Any other type encountered is transferred to TPESORT for processing. When the Additions are completed, the Changes are processed from the tape created by TPESORT. Then Deletions and finally Error records are processed in the same manner. Printing of the data is handled by PRINT, which also handles the physical spacing and page ejections.

Two types of data are listed: searchable segments and free text segments. The searchable segments are handled first. They include a header and a number of descriptors, sub-descriptors, and subordinate data pertaining to each. The various descriptors are printed along with their sub-descriptors and subordinate data beneath the header. If a free text segment follows it is printed beneath the descriptor data. All binary numbers encountered are converted to decimal in CVDROT prior to printing. As each tape being processed reaches an end of file, control is transferred to the end of file routine for that particular file. (see END-OF-FILE ROUTINES).

2. TPESORT Routine

On the first pass any non-Addition type record is transferred to this routine. The type of record is determined and the record is written on the particular work tape reserved for it. These work tapes later become input to MPRINT1.

3. Print Routine

As soon as the output area is full PRINT is entered. A test is made for an end of page condition and a page is ejected and titled if necessary. Then the buffer is printed and cleared and control is returned to the main program.

CVDROT Routine

Since binary numbers cannot be printed they must be converted to decimal. CVDROT converts the number to decimal and tests the leading (left-most) digits. As soon as a non-zero is encountered, (or, in the case of the number 1, the last digit), the remaining digits are returned to the main program in decimal form.

End of File Routines

1. MASTAPE

When the end of file is encountered on the input tape from the Master File Maintenance Program the work tape files are closed and an input file for CHNTAPE(changes) is opened. If it is determined that no changes are requested control is transferred to CHNTAPE end of file.

2. CHNTAPE

When this routine is entered the CHNTAPE file is closed and the DELTAPE(deletions) input file is opened. If it is determined that no Deletions are requested control is transferred to DELTAPE end of file.

3. DELTAPE

When this routine is entered, the DELTAPE file is closed and the ERRTAPE(errors) input file is opened. If it is determined that no Errors are requested control is transferred to ERRTAPE end of file.

4. ERRTAPE

When this routine is entered the ERRTAPE file is closed and processing terminates.

Input

The tape produced by MFMAINT containing records which have been altered is read in the format (M2) shown below.

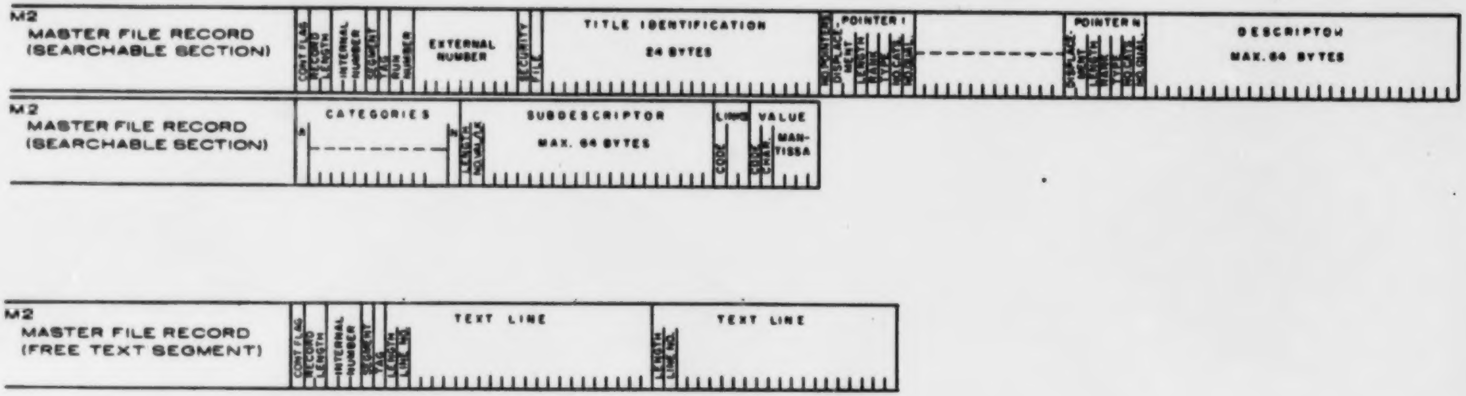
Output

MPRINT1 prints the reformatted records in the printer format shown below. The ID. file is printed by a slightly modified version of MPRINT1 (MPRINT2). The basic difference is the wording used in the header.

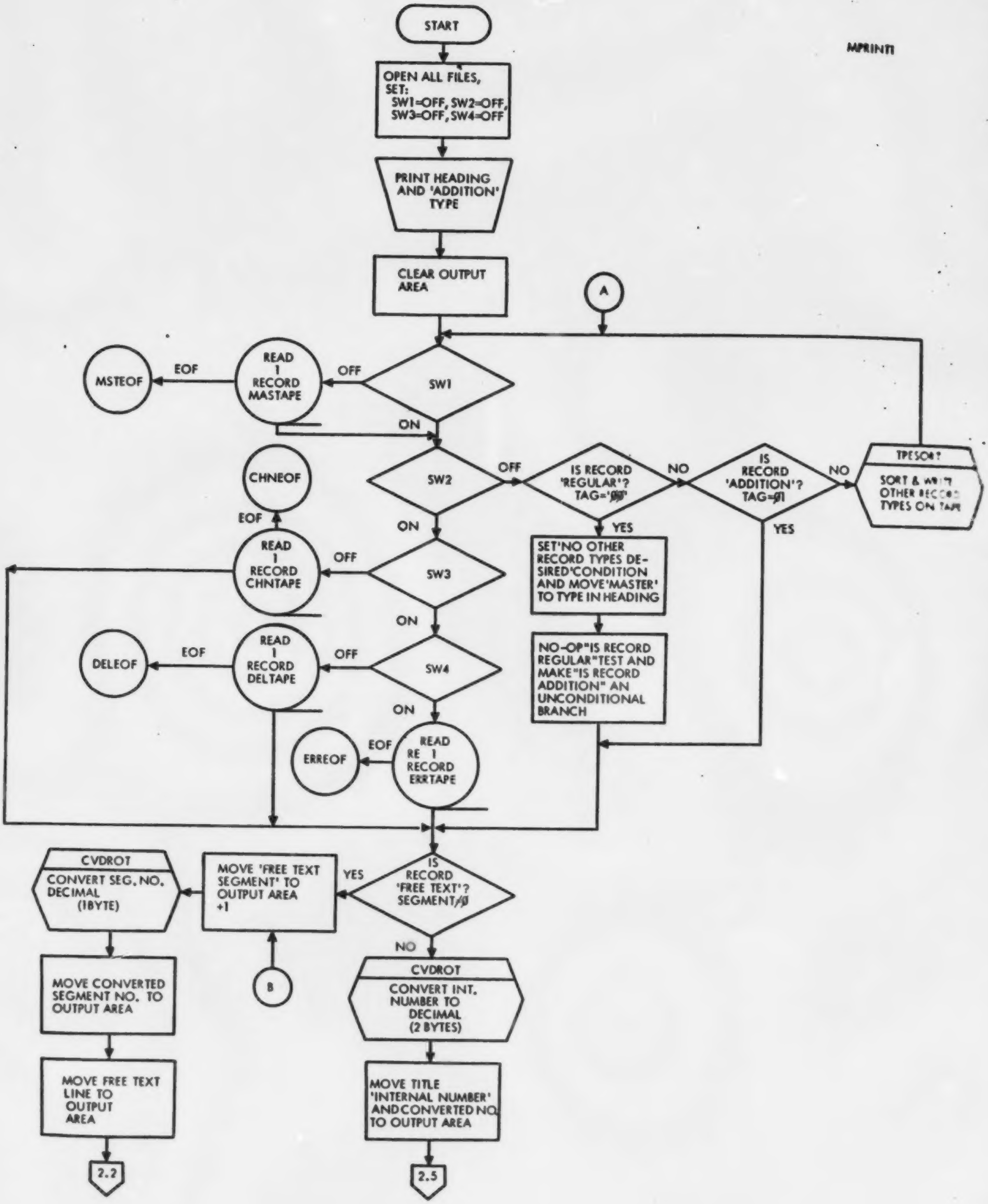
```
MASTER FILE MAINTENANCE LISTING                MM/DD/YY
            ADDITIONS
            CHANGES
            DELETIONS
            ERRORS


INTERNAL NUMBER, 12345678, EXTERNAL DOCUMENT NUMBER 1234567, FILE 123, IDENTIFICATION ABCDEFGHIJKLMNOPQRSTUVW
DRRR/DESCRIPTOR, C/CAT, C/CAT...C/CAT, LK/(L1,L2,L3), VS/NUMBER, VL/NUMBER, VU/NUMBER,...S/SUBDESCRIPTOR


INTERNAL NUMBER 12345678, EXTERNAL DOCUMENT NUMBER 1234567, FILE 123, IDENTIFICATION A...
  FREE TEXT SEGMENT 123          TEXT LINE
                       LINE XXX
```

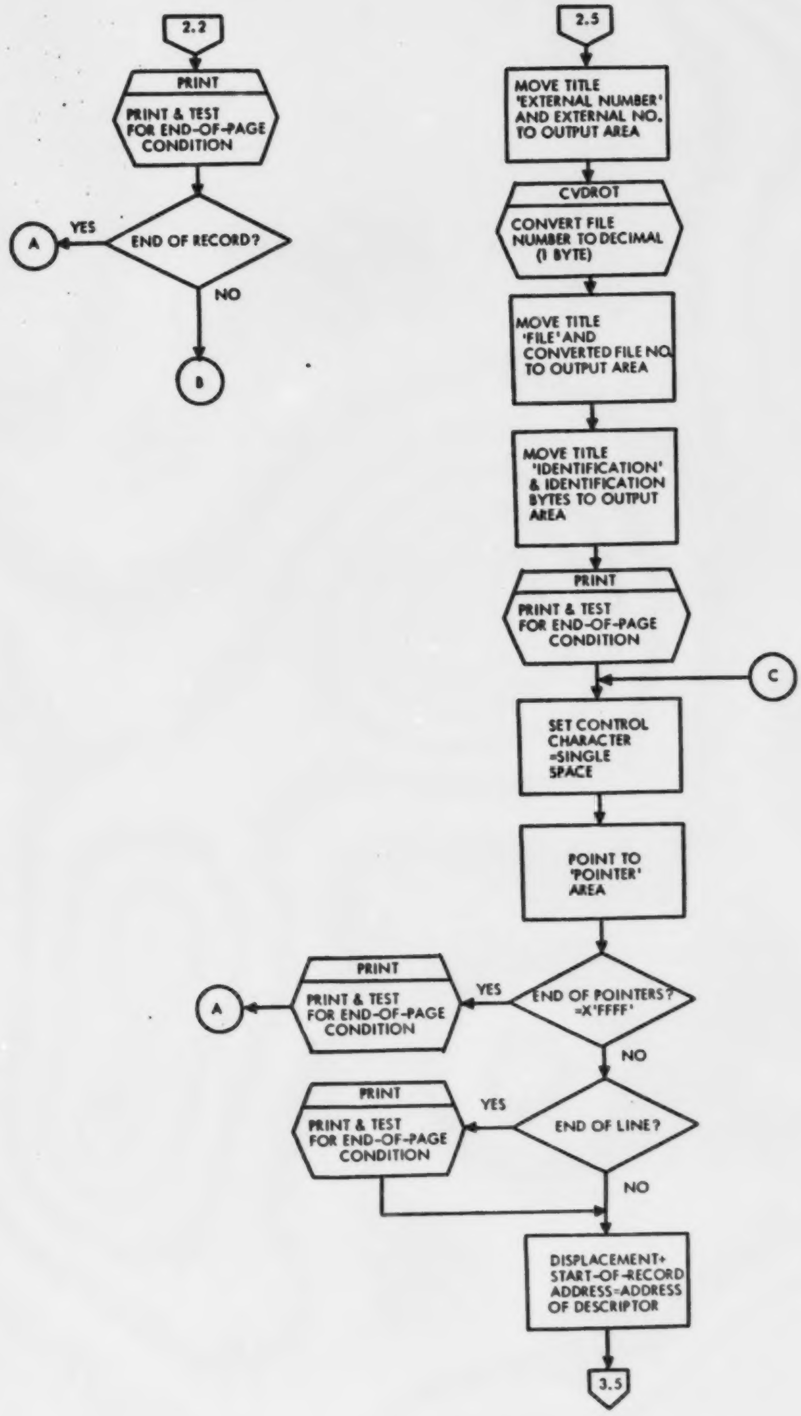
MPRINT1
OUTPUT FORMAT

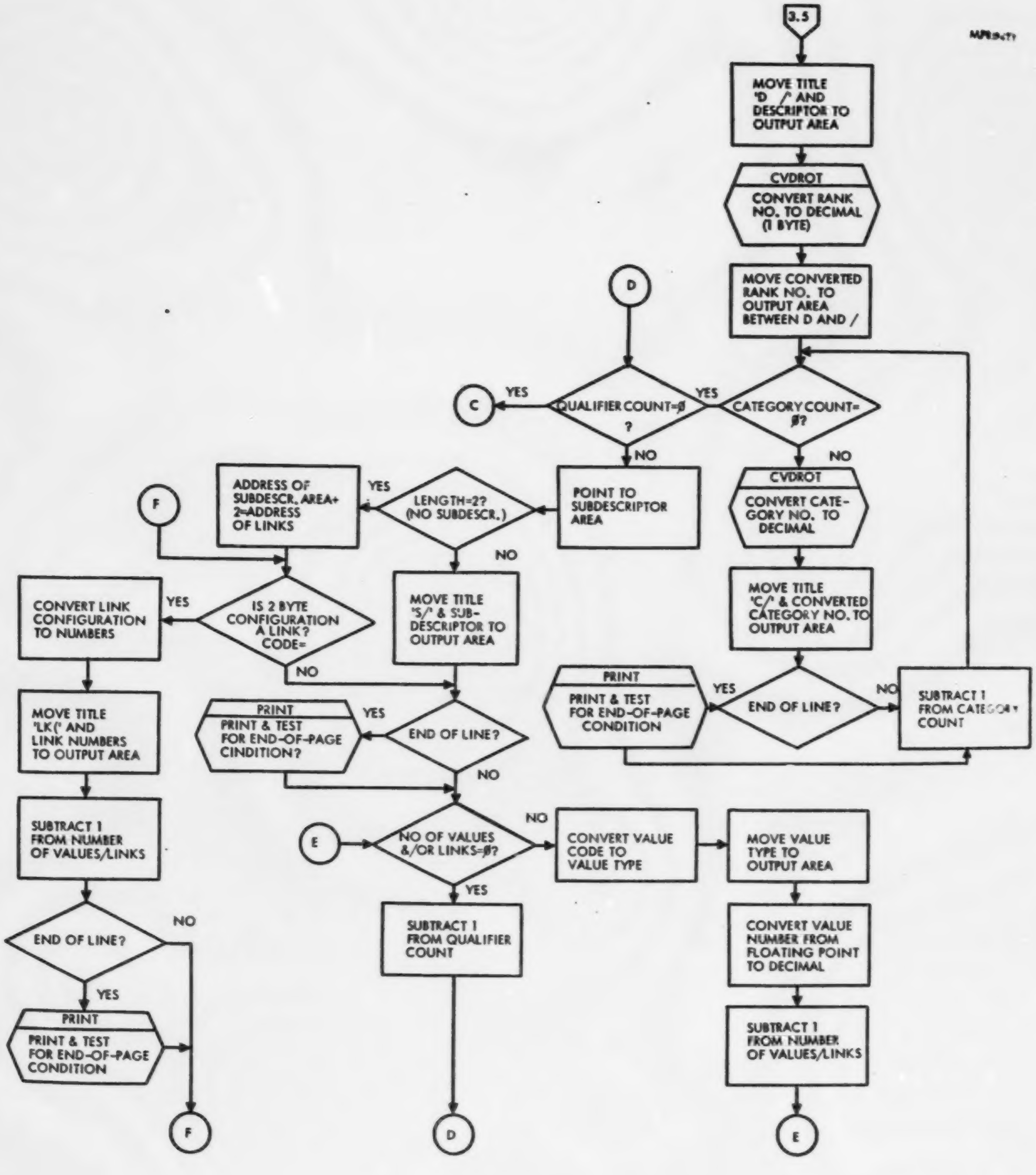
MASTER FILE PRINT PROGRAM (MPRINT1) RECORD FORMATS


M2
MASTER FILE RECORD
(SEARCHABLE SECTION)
CONT FLAG
RECORD LENGTH
INTERNAL NUMBER
SEGMENT
TAG
RUN NUMBER
EXTERNAL NUMBER
SECURITY
FILE
TITLE IDENTIFICATION
24 BYTES
NO. POINTERS
POINTER 1
DISPLACE-MENT
LENGTH
RANK
TYPE
NO.CATS
NO.QUAL
POINTER N
DESCRIPTOR
MAX. 64 BYTES
M2
MASTER FILE RECORD
(SEARCHABLE SECTION)
CATEGORIES
SUBDESCRIPTOR
MAX. 64 BYTES
LENGTH
NO.VALUE
LINK
CODE
VALUE
CODE
CHAR.
MAN-TISSA
M2
MASTER FILE RECORD
(FREE TEXT SEGMENT)
CONT FLAG
RECORD LENGTH
INTERNAL NUMBER
SEGMENT
TAG
LENGTH
LINE NO.
TEXT LINE
LENGTH
LINE NO.
TEXT LINE

MPRINT1

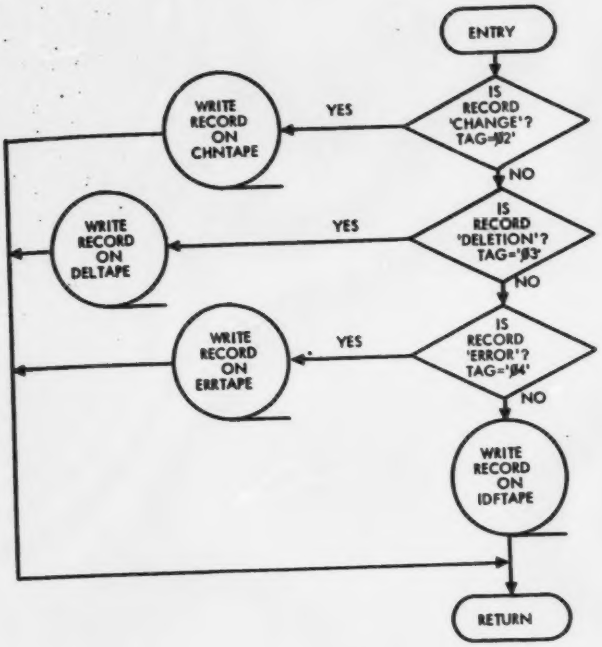

START
OPEN ALL FILES, SET: SW1=OFF, SW2=OFF, SW3=OFF, SW4=OFF
PRINT HEADING AND 'ADDITION' TYPE
CLEAR OUTPUT AREA
A
SW1
OFF
READ 1 RECORD MASTAPE
EOF
MSTEOF
ON
SW2
OFF
IS RECORD 'REGULAR'? TAG='00'
NO
IS RECORD 'ADDITION'? TAG=01
NO
TPESORT
SORT & WRITE OTHER RECORD TYPES ON TAPE
YES
SET 'NO OTHER RECORD TYPES DESIRED' CONDITION AND MOVE 'MASTER' TO TYPE IN HEADING
NO-OP "IS RECORD REGULAR" TEST AND MAKE "IS RECORD ADDITION" AN UNCONDITIONAL BRANCH
YES
ON
CHNEOF
EOF
READ 1 RECORD CHNTAPE
OFF
SW3
ON
READ 1 RECORD DELTAPE
EOF
DELEOF
OFF
SW4
ON
READ RE 1 RECORD ERRTAPE
EOF
ERREOF
IS RECORD 'FREE TEXT'? SEGMENT≠0
YES
MOVE 'FREE TEXT SEGMENT' TO OUTPUT AREA +1
CVDROT
CONVERT SEG. NO. DECIMAL (1BYTE)
MOVE CONVERTED SEGMENT NO. TO OUTPUT AREA
MOVE FREE TEXT LINE TO OUTPUT AREA
2.2
B
NO
CVDROT
CONVERT INT. NUMBER TO DECIMAL (2 BYTES)
MOVE TITLE 'INTERNAL NUMBER' AND CONVERTED NO. TO OUTPUT AREA
2.5


2.2
PRINT
PRINT & TEST FOR END-OF-PAGE CONDITION
END OF RECORD?
YES
A
NO
B
2.5
MOVE TITLE 'EXTERNAL NUMBER' AND EXTERNAL NO. TO OUTPUT AREA
CVDROT
CONVERT FILE NUMBER TO DECIMAL (1 BYTE)
MOVE TITLE 'FILE' AND CONVERTED FILE NO. TO OUTPUT AREA
MOVE TITLE 'IDENTIFICATION' & IDENTIFICATION BYTES TO OUTPUT AREA
PRINT
PRINT & TEST FOR END-OF-PAGE CONDITION
C
SET CONTROL CHARACTER =SINGLE SPACE
POINT TO 'POINTER' AREA
END OF POINTERS? =X'FFFF'
YES
PRINT
PRINT & TEST FOR END-OF-PAGE CONDITION
A
NO
END OF LINE?
YES
PRINT
PRINT & TEST FOR END-OF-PAGE CONDITION
NO
DISPLACEMENT+ START-OF-RECORD ADDRESS=ADDRESS OF DESCRIPTOR
3.5

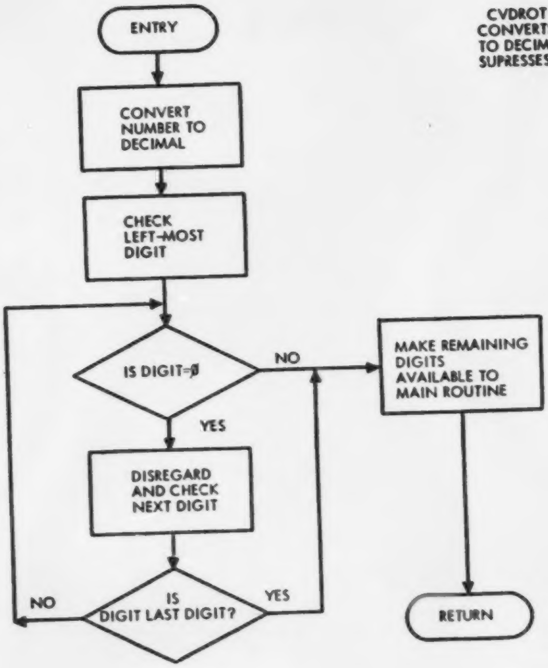

3.5
MOVE TITLE 'D /' AND DESCRIPTOR TO OUTPUT AREA
CVDROT
CONVERT RANK NO. TO DECIMAL (1 BYTE)
MOVE CONVERTED RANK NO. TO OUTPUT AREA BETWEEN D AND /
D
QUALIFIER COUNT=0 ?
YES
C
YES
CATEGORY COUNT= 0?
NO
NO
POINT TO SUBDESCRIPTOR AREA
LENGTH=2? (NO SUBDESCR.)
YES
ADDRESS OF SUBDESCR. AREA+ 2=ADDRESS OF LINKS
NO
F
CVDROT
CONVERT CATEGORY NO. TO DECIMAL
MOVE TITLE 'C/' & CONVERTED CATEGORY NO. TO OUTPUT AREA
IS 2 BYTE CONFIGURATION A LINK? CODE=
YES
CONVERT LINK CONFIGURATION TO NUMBERS
NO
MOVE TITLE 'S/' & SUBDESCRIPTOR TO OUTPUT AREA
PRINT
PRINT & TEST FOR END-OF-PAGE CONDITION
YES
END OF LINE?
NO
SUBTRACT 1 FROM CATEGORY COUNT
MOVE TITLE 'LK(' AND LINK NUMBERS TO OUTPUT AREA
PRINT
PRINT & TEST FOR END-OF-PAGE CINDITION?
YES
END OF LINE?
NO
SUBTRACT 1 FROM NUMBER OF VALUES/LINKS
E
NO OF VALUES &/OR LINKS=0?
NO
CONVERT VALUE CODE TO VALUE TYPE
MOVE VALUE TYPE TO OUTPUT AREA
YES
SUBTRACT 1 FROM QUALIFIER COUNT
CONVERT VALUE NUMBER FROM FLOATING POINT TO DECIMAL
END OF LINE?
NO
YES
PRINT
PRINT & TEST FOR END-OF-PAGE CONDITION
SUBTRACT 1 FROM NUMBER OF VALUES/LINKS
F
D
E

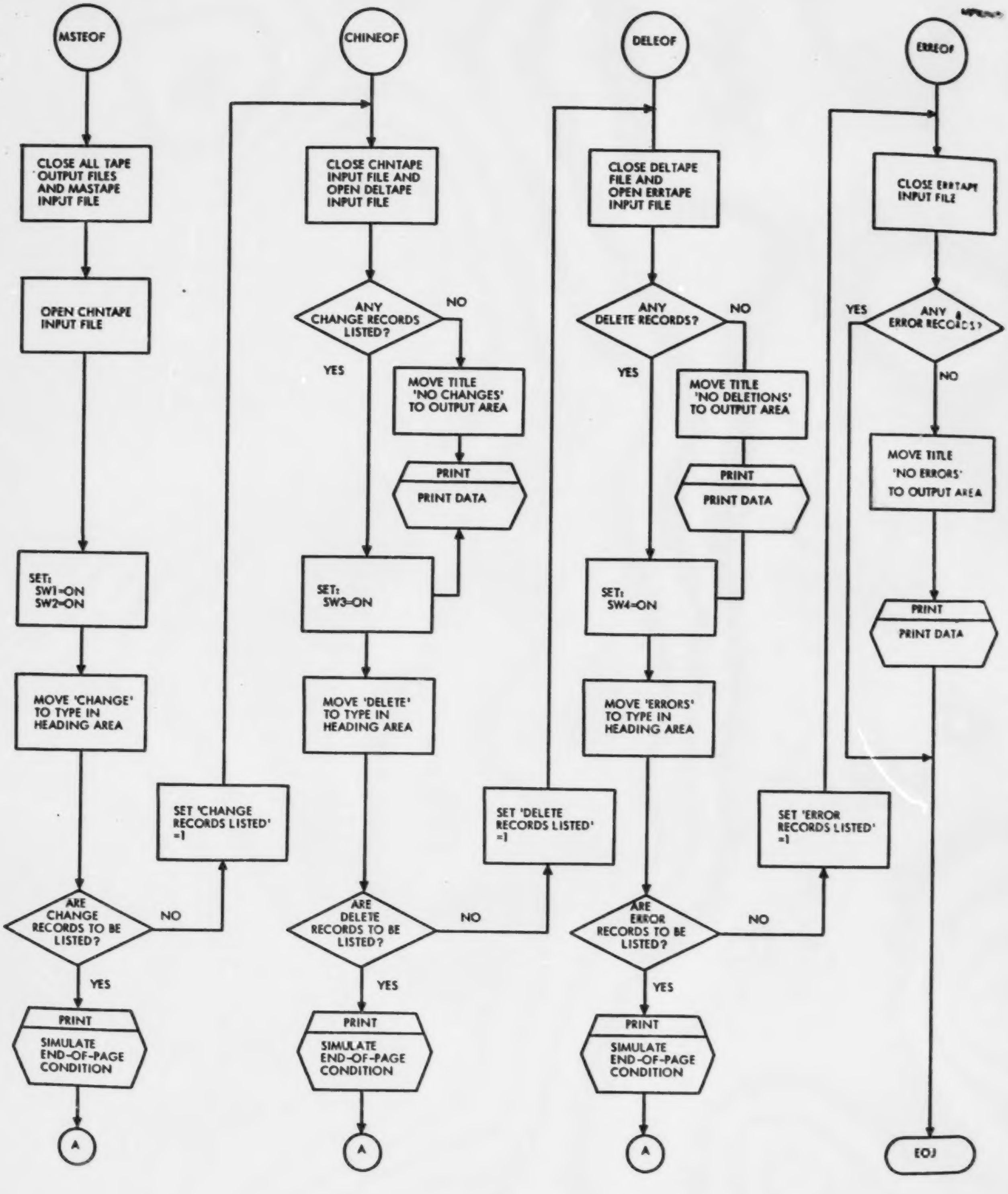
MPRINTI

TPESORT


ENTRY
IS RECORD 'CHANGE'? TAG='02'
YES
WRITE RECORD ON CHNTAPE
NO
IS RECORD 'DELETION'? TAG='03'
YES
WRITE RECORD ON DELTAPE
NO
IS RECORD 'ERROR'? TAG='04'
YES
WRITE RECORD ON ERRTAPE
NO
WRITE RECORD ON IDFTAPE
RETURN

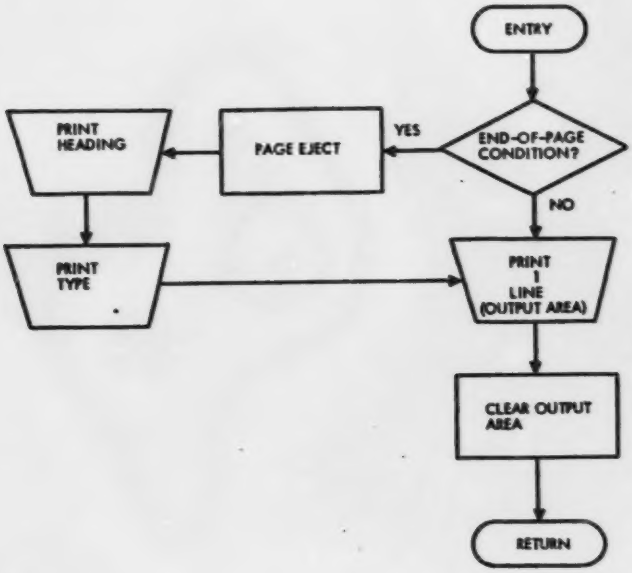
CVDROT
CONVERTS NOS
TO DECIMAL &
SUPRESSES LEAD 0'


ENTRY
CONVERT NUMBER TO DECIMAL
CHECK LEFT-MOST DIGIT
IS DIGIT=0
NO
YES
MAKE REMAINING DIGITS AVAILABLE TO MAIN ROUTINE
DISREGARD AND CHECK NEXT DIGIT
IS DIGIT LAST DIGIT?
NO
YES
RETURN


MSTEOF
CLOSE ALL TAPE OUTPUT FILES AND MASTAPE INPUT FILE
OPEN CHNTAPE INPUT FILE
SET: SW1=ON SW2=ON
MOVE 'CHANGE' TO TYPE IN HEADING AREA
ARE CHANGE RECORDS TO BE LISTED?
NO
SET 'CHANGE RECORDS LISTED' =1
YES
PRINT
SIMULATE END-OF-PAGE CONDITION
A
CHINEOF
CLOSE CHNTAPE INPUT FILE AND OPEN DELTAPE INPUT FILE
ANY CHANGE RECORDS LISTED?
NO
YES
MOVE TITLE 'NO CHANGES' TO OUTPUT AREA
PRINT
PRINT DATA
SET: SW3=ON
MOVE 'DELETE' TO TYPE IN HEADING AREA
ARE DELETE RECORDS TO BE LISTED?
NO
SET 'DELETE RECORDS LISTED' =1
YES
PRINT
SIMULATE END-OF-PAGE CONDITION
A
DELEOF
CLOSE DELTAPE FILE AND OPEN ERRTAPE INPUT FILE
ANY DELETE RECORDS?
NO
YES
MOVE TITLE 'NO DELETIONS' TO OUTPUT AREA
PRINT
PRINT DATA
SET: SW4=ON
MOVE 'ERRORS' TO TYPE IN HEADING AREA
ARE ERROR RECORDS TO BE LISTED?
NO
SET 'ERROR RECORDS LISTED' =1
YES
PRINT
SIMULATE END-OF-PAGE CONDITION
A
ERREOF
CLOSE ERRTAPE INPUT FILE
ANY ERROR RECORDS?
YES
NO
MOVE TITLE 'NO ERRORS' TO OUTPUT AREA
PRINT
PRINT DATA
EOJ

MPRINT1

PRINT


ENTRY
END-OF-PAGE CONDITION?
YES
NO
PAGE EJECT
PRINT HEADING
PRINT TYPE
PRINT 1 LINE (OUTPUT AREA)
CLEAR OUTPUT AREA
RETURN

ID-FILE PRINT PROGRAM (MPRINT2) PROCESSING LOGIC

MPRINT2 uses the same processing logic as MPRINT1 except for putting different page headings and deblocking the ID file records.

DATA CELL DUMP PROGRAM - BLOCK CHANGE EDIT PHASE (EDITBC) PROCESSING LOGIC

EDITBC reads an input card and checks the format for errors. If none are found the card type (replace or delete) is determined. If the card is a Delete the term is moved to the output area, the Delete code is moved in, and the next card is read. In the case of a Replace, the term and its replacement are moved to the output area (with the length of each) and the Replace code is moved in. Since a Replace card may be continued, the next card is tested. If it is a continuation card, the remainder of data is moved to the output area. If it is not a continuation card, it is processed as above.

When an end of file occurs on the input device, the data in the output area is sorted and passed to the next phase.

DATA CELL DUMP PROGRAM - DUMP PHASE (DUMPCELL) PROCESSING LOGIC

This phase reads a record from the data cell and checks if delete tag is set. If so it reads the next record and checks if there are any block changes to be made. If so each descriptor and/or subdescriptor in the record is searched in the block change table. If found it either deletes or replaces it with the term in the table. After the whole record is scanned and block changes performed, it writes it onto a blocked master file tape. This process is repeated until there are no more records on the data cell.

As a result of the block changes, input records are created to the Inverted file, X-Ref file and ID-file update programs.

Input

Block change records are stored in the format shown below labelled M7. The master file records (M2) are read from the data cell.

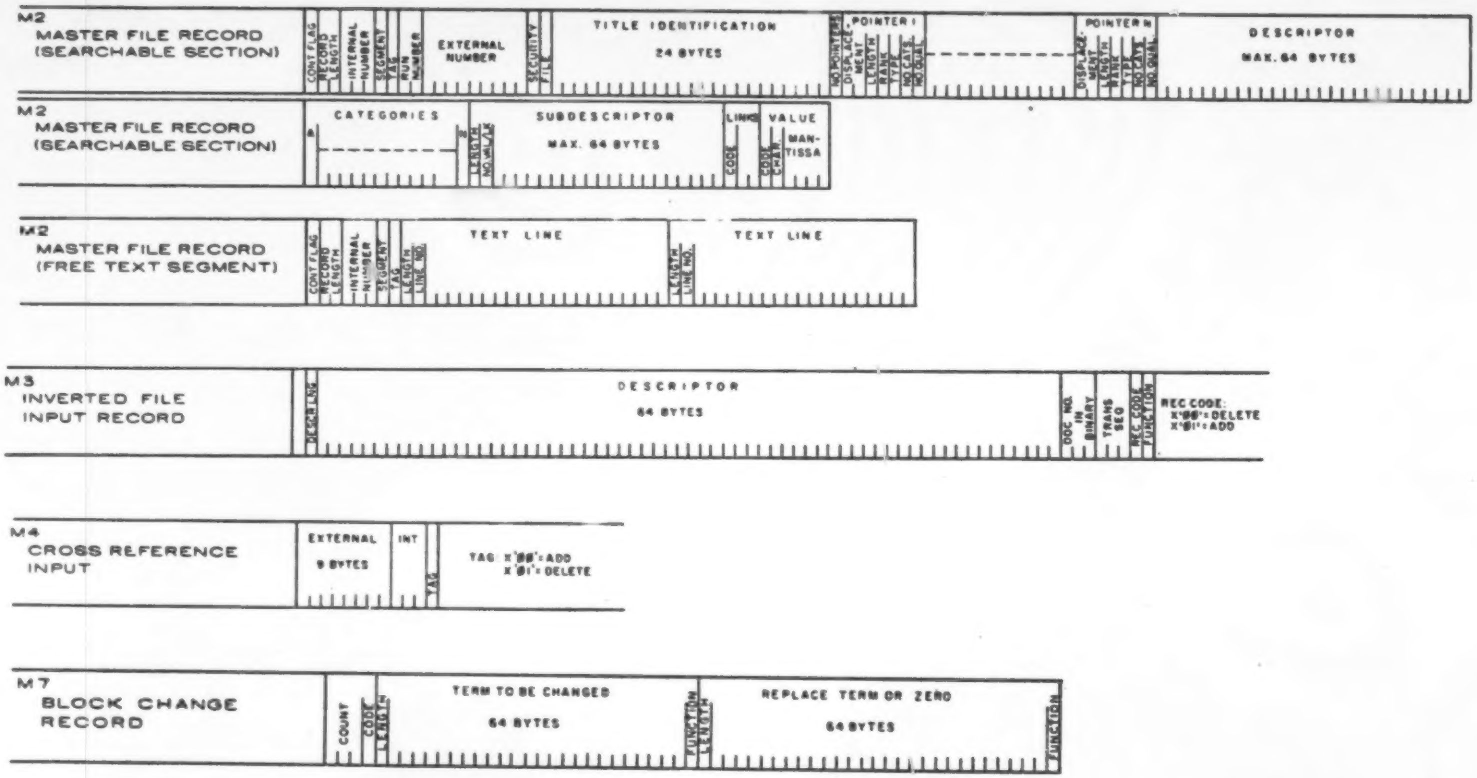
Output

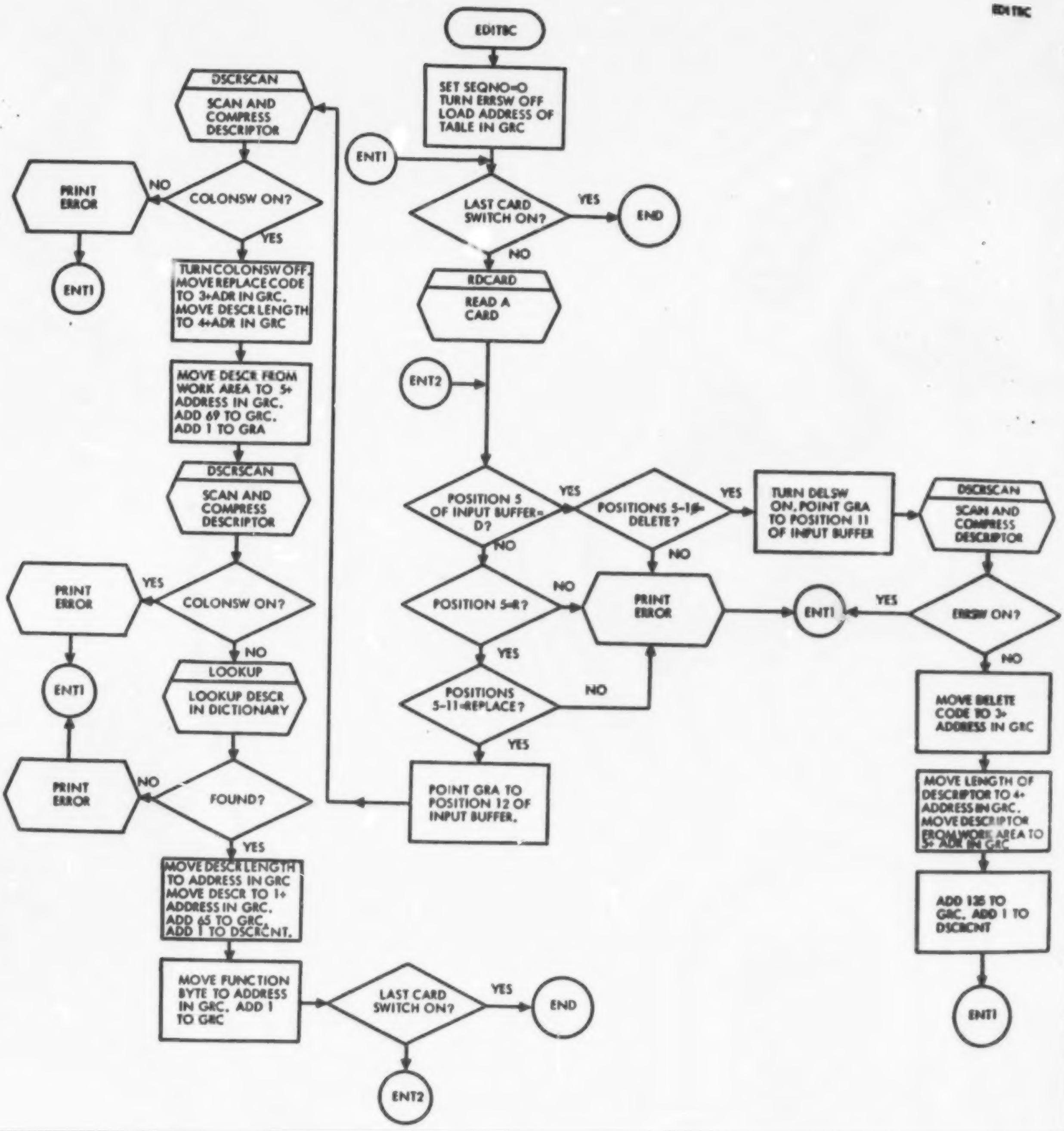
A tape version of the entire master file is created in the M2 format below. .For a more detailed description of the master file record, please refer to Appendix B.

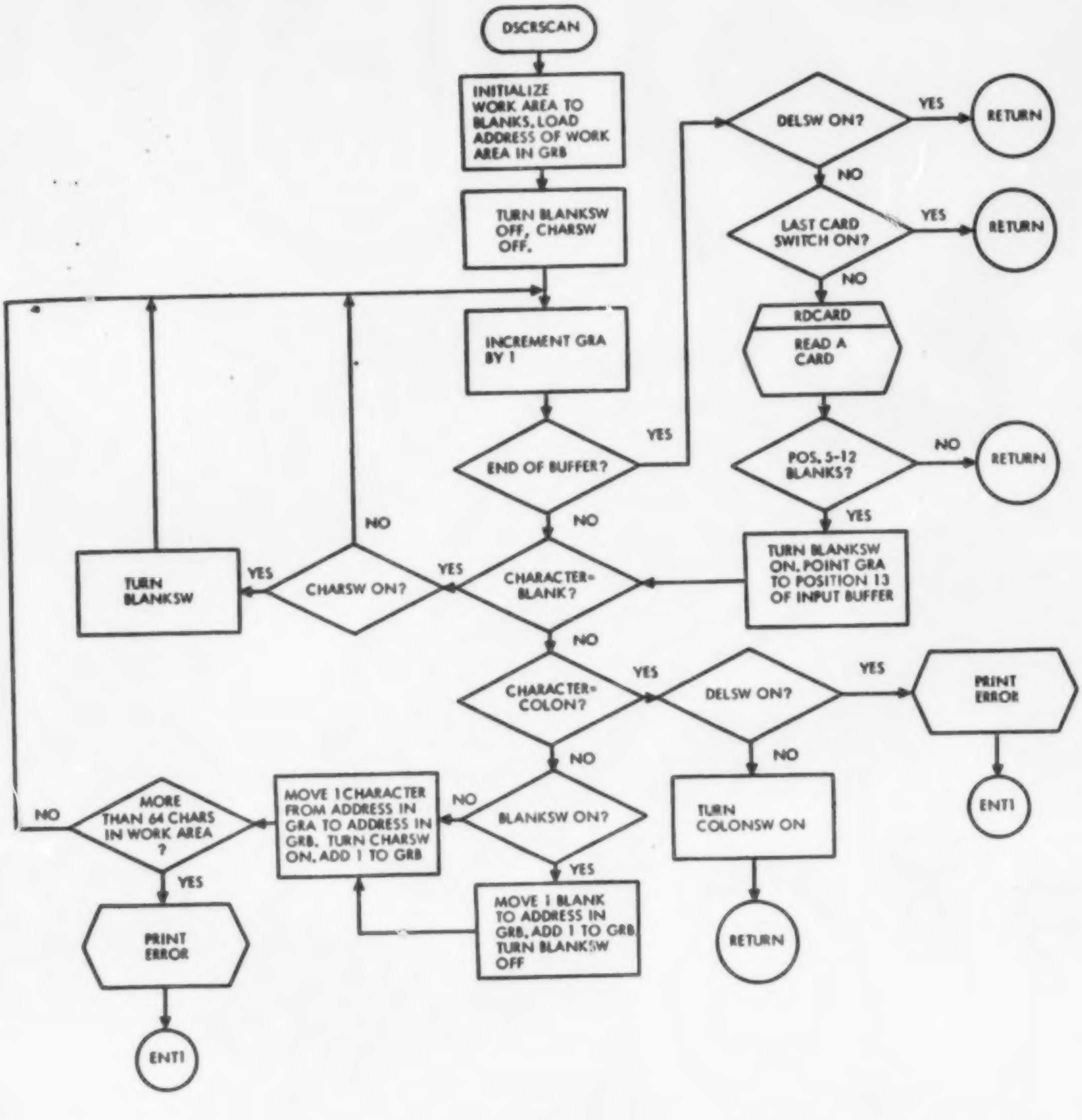
In addition, update tapes are created to the subordinate files. These three tapes are:

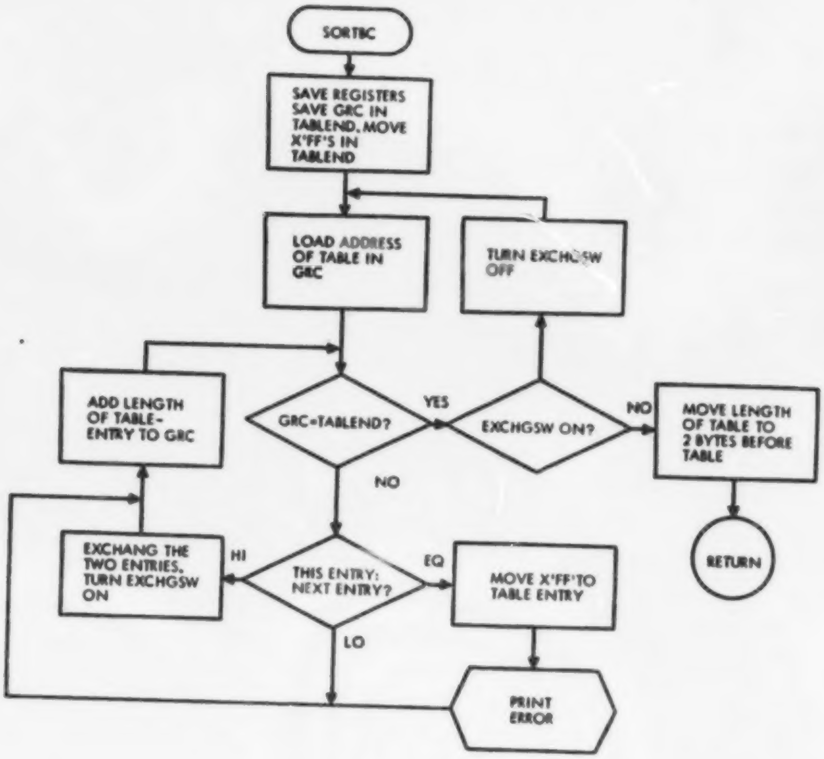
1. Inverted file update (format M3)
2. Cross reference update (format M4)
3. ID. file update (format M2)

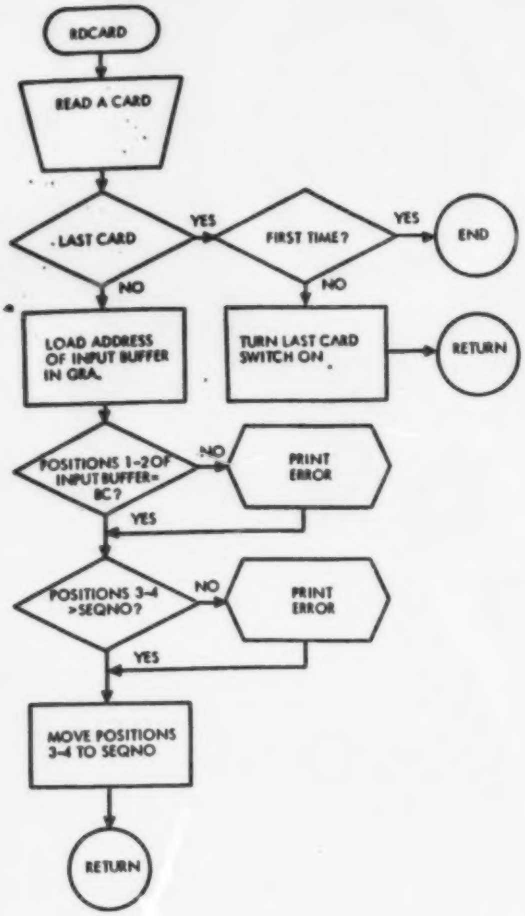
DATA CELL DUMP PROGRAM (DUMPCELL) RECORD FORMATS

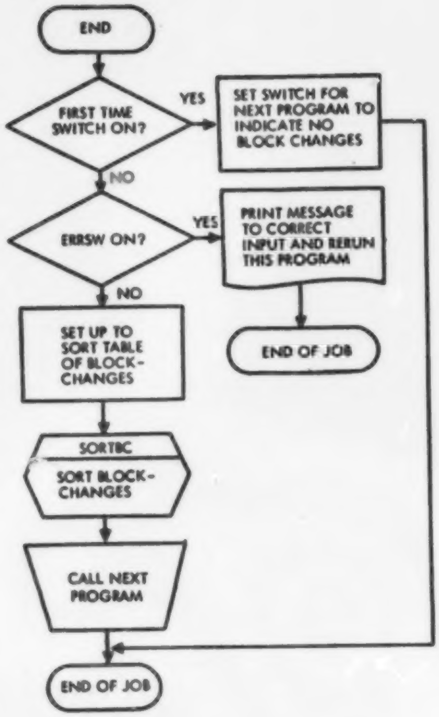

M2
MASTER FILE RECORD
(SEARCHABLE SECTION)
CONT FLAG
RECORD LENGTH
INTERNAL NUMBER
SEGMENT
TAG
RUN NUMBER
EXTERNAL NUMBER
SECURITY
FILE
TITLE IDENTIFICATION
24 BYTES
NO POINTERS
POINTER I
DISPLACEMENT
LENGTH
RANK
TYPE
NO CATS
NO QUAL
POINTER N
DISPLACEMENT
LENGTH
RANK
TYPE
NO CATS
NO QUAL
DESCRIPTOR
MAX. 64 BYTES
M2
MASTER FILE RECORD
(SEARCHABLE SECTION)
CATEGORIES
A
N
LENGTH
NO VAL/LK
SUBDESCRIPTOR
MAX. 64 BYTES
LINKS
CODE
VALUE
CODE
CHAR.
MANTISSA
M2
MASTER FILE RECORD
(FREE TEXT SEGMENT)
CONT FLAG
RECORD LENGTH
INTERNAL NUMBER
SEGMENT
TAG
LENGTH
LINE NO.
TEXT LINE
LENGTH
LINE NO.
TEXT LINE
M3
INVERTED FILE
INPUT RECORD
DESCR LNG
DESCRIPTOR
64 BYTES
DOC NO. IN BINARY
TRANS SEQ
REC CODE
FUNCTION
REC CODE:
X'00':DELETE
X'01':ADD
M4
CROSS REFERENCE
INPUT
EXTERNAL
9 BYTES
INT
TAG
TAG: X'00':ADD
X'01':DELETE
M7
BLOCK CHANGE
RECORD
COUNT
CODE
LENGTH
TERM TO BE CHANGED
64 BYTES
FUNCTION
LENGTH
REPLACE TERM OR ZERO
64 BYTES
FUNCTION

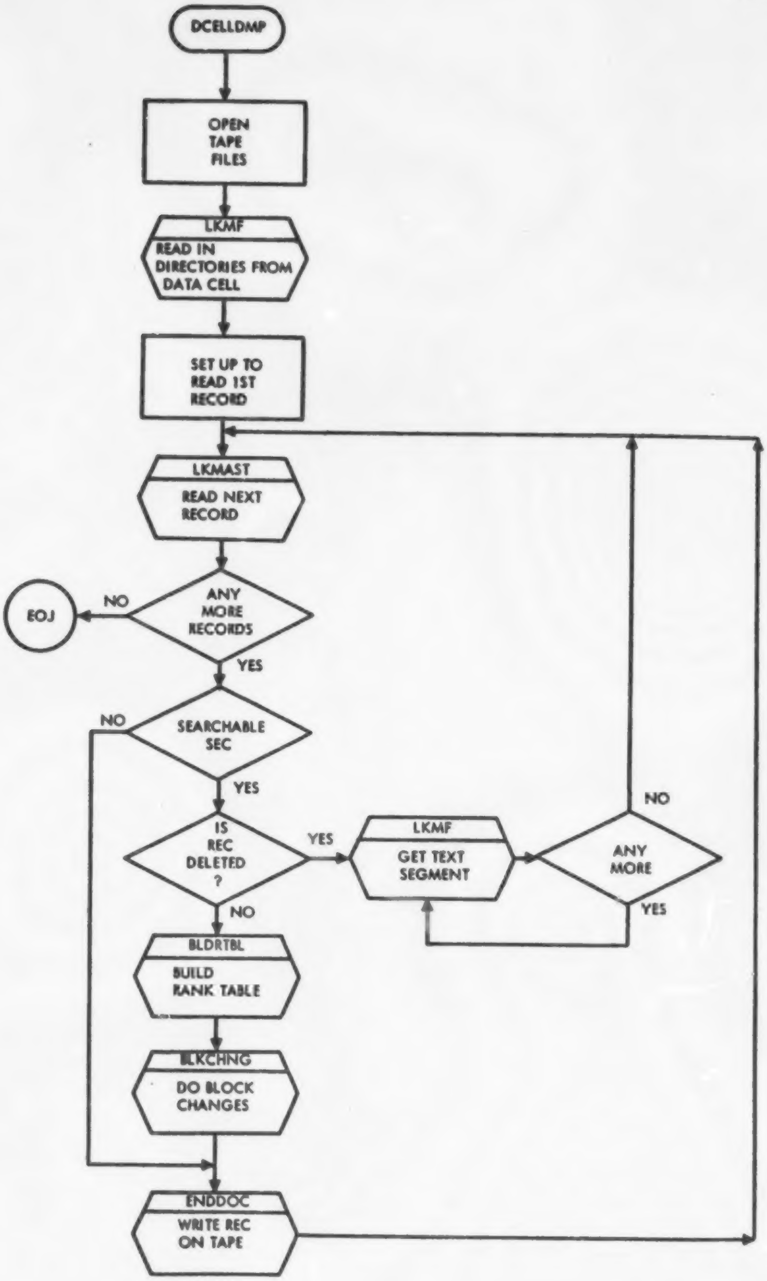

EDITBC
SET SEQNO=O
TURN ERRSW OFF
LOAD ADDRESS OF
TABLE IN GRC
ENT1
LAST CARD SWITCH ON?
YES
END
NO
RDCARD
READ A CARD
ENT2
POSITION 5 OF INPUT BUFFER=D?
YES
POSITIONS 5-10=DELETE?
YES
TURN DELSW ON, POINT GRA TO POSITION 11 OF INPUT BUFFER
DSCRSCAN
SCAN AND COMPRESS DESCRIPTOR
NO
NO
POSITION 5=R?
NO
PRINT ERROR
ENT1
YES
ERRSW ON?
NO
YES
POSITIONS 5-11=REPLACE?
NO
YES
POINT GRA TO POSITION 12 OF INPUT BUFFER.
MOVE DELETE CODE TO 3+ ADDRESS IN GRC
MOVE LENGTH OF DESCRIPTOR TO 4+ ADDRESS IN GRC. MOVE DESCRIPTOR FROM WORK AREA TO 5+ ADR IN GRC
ADD 125 TO GRC. ADD 1 TO DSCRCNT
ENT1
DSCRSCAN
SCAN AND COMPRESS DESCRIPTOR
COLONSW ON?
NO
PRINT ERROR
ENT1
YES
TURN COLONSW OFF. MOVE REPLACE CODE TO 3+ADR IN GRC. MOVE DESCR LENGTH TO 4+ADR IN GRC
MOVE DESCR FROM WORK AREA TO 5+ ADDRESS IN GRC. ADD 69 TO GRC. ADD 1 TO GRA
DSCRSCAN
SCAN AND COMPRESS DESCRIPTOR
COLONSW ON?
YES
PRINT ERROR
ENT1
NO
LOOKUP
LOOKUP DESCR IN DICTIONARY
FOUND?
NO
PRINT ERROR
YES
MOVE DESCR LENGTH TO ADDRESS IN GRC MOVE DESCR TO 1+ ADDRESS IN GRC. ADD 65 TO GRC. ADD 1 TO DSCRCNT.
MOVE FUNCTION BYTE TO ADDRESS IN GRC. ADD 1 TO GRC
LAST CARD SWITCH ON?
YES
END
ENT2

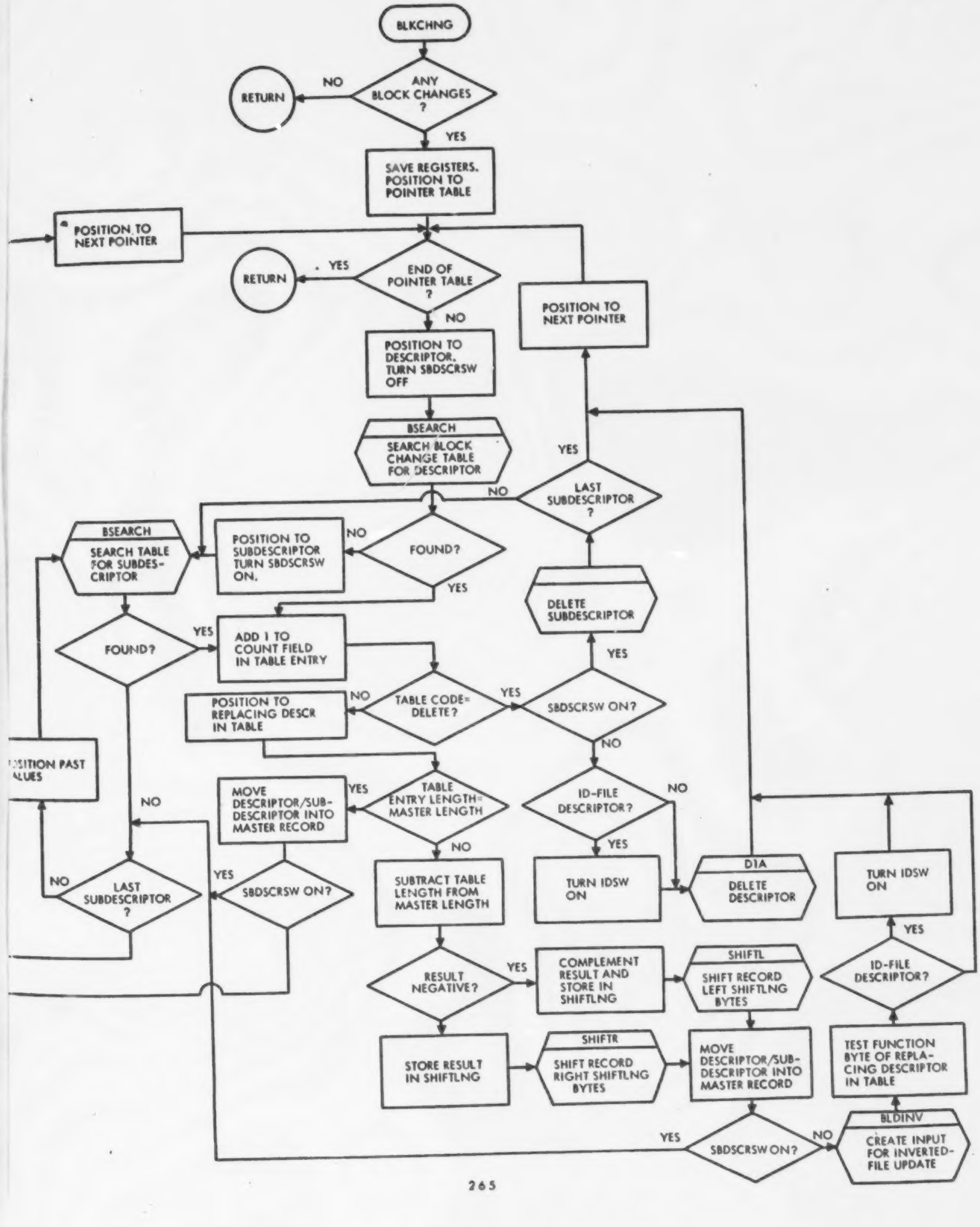

DSCRSCAN
INITIALIZE WORK AREA TO BLANKS, LOAD ADDRESS OF WORK AREA IN GRB
TURN BLANKSW OFF, CHARSW OFF.
INCREMENT GRA BY 1
END OF BUFFER?
YES
DELSW ON?
YES
RETURN
NO
LAST CARD SWITCH ON?
YES
RETURN
NO
RDCARD
READ A CARD
POS. 5-12 BLANKS?
NO
RETURN
YES
TURN BLANKSW ON, POINT GRA TO POSITION 13 OF INPUT BUFFER
NO
CHARACTER= BLANK?
YES
CHARSW ON?
NO
YES
TURN BLANKSW
NO
CHARACTER= COLON?
YES
DELSW ON?
YES
PRINT ERROR
ENT1
NO
TURN COLONSW ON
RETURN
NO
BLANKSW ON?
NO
MOVE 1 CHARACTER FROM ADDRESS IN GRA TO ADDRESS IN GRB. TURN CHARSW ON, ADD 1 TO GRB
YES
MOVE 1 BLANK TO ADDRESS IN GRB, ADD 1 TO GRB TURN BLANKSW OFF
MORE THAN 64 CHARS IN WORK AREA ?
NO
YES
PRINT ERROR
ENT1

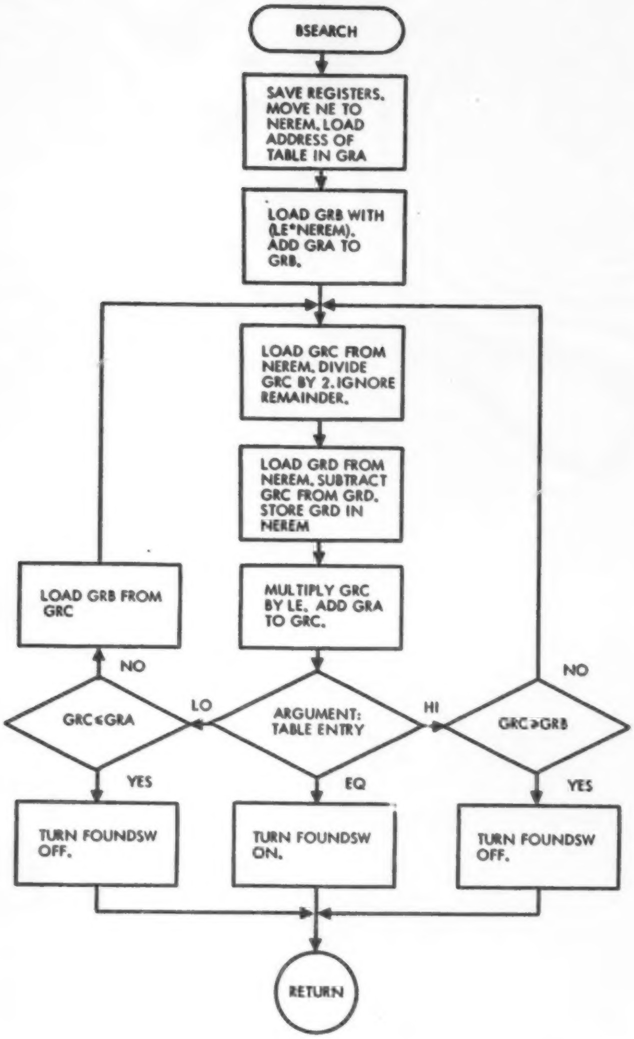

SORTBC
SAVE REGISTERS SAVE GRC IN TABLEND, MOVE X'FF'S IN TABLEND
LOAD ADDRESS OF TABLE IN GRC
TURN EXCHGSW OFF
ADD LENGTH OF TABLE-ENTRY TO GRC
GRC=TABLEND?
YES
EXCHGSW ON?
NO
MOVE LENGTH OF TABLE TO 2 BYTES BEFORE TABLE
NO
RETURN
EXCHANG THE TWO ENTRIES, TURN EXCHGSW ON
HI
THIS ENTRY: NEXT ENTRY?
EQ
MOVE X'FF'TO TABLE ENTRY
LO
PRINT ERROR

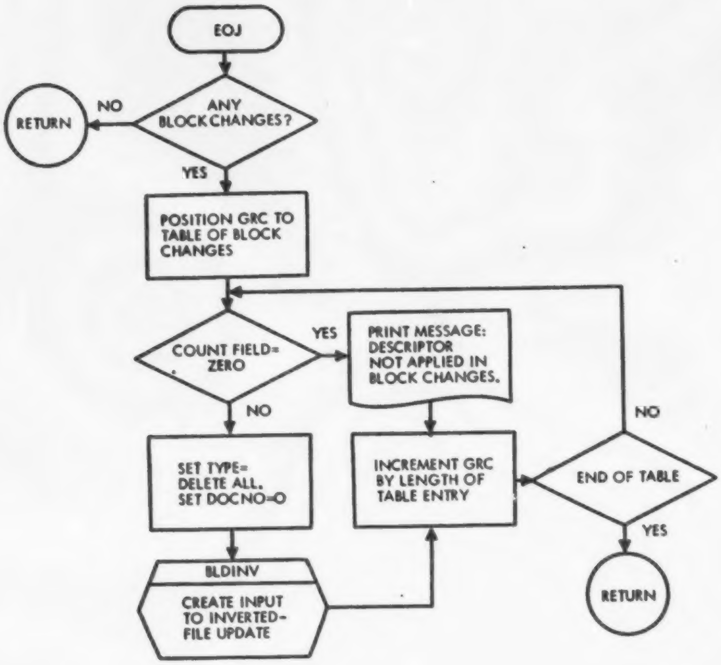

RDCARD
READ A CARD
LAST CARD
YES
FIRST TIME?
YES
END
NO
NO
LOAD ADDRESS OF INPUT BUFFER IN GRA.
TURN LAST CARD SWITCH ON
RETURN
POSITIONS 1-2 OF INPUT BUFFER= BC?
NO
PRINT ERROR
YES
POSITIONS 3-4 >SEQNO?
NO
PRINT ERROR
YES
MOVE POSITIONS 3-4 TO SEQNO
RETURN

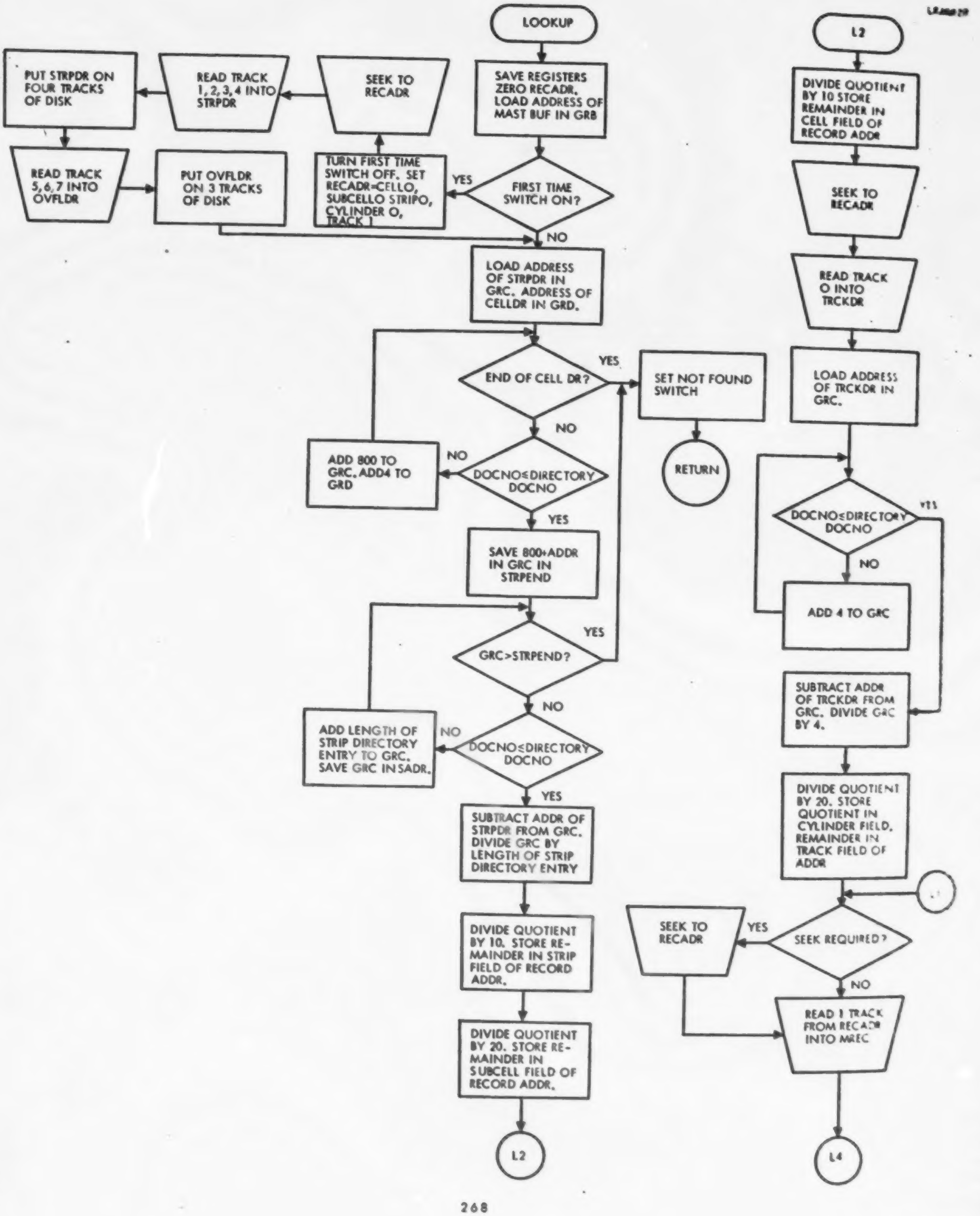

END
FIRST TIME SWITCH ON?
YES
SET SWITCH FOR NEXT PROGRAM TO INDICATE NO BLOCK CHANGES
NO
ERRSW ON?
YES
PRINT MESSAGE TO CORRECT INPUT AND RERUN THIS PROGRAM
END OF JOB
NO
SET UP TO SORT TABLE OF BLOCK-CHANGES
SORTBC
SORT BLOCK-CHANGES
CALL NEXT PROGRAM
END OF JOB

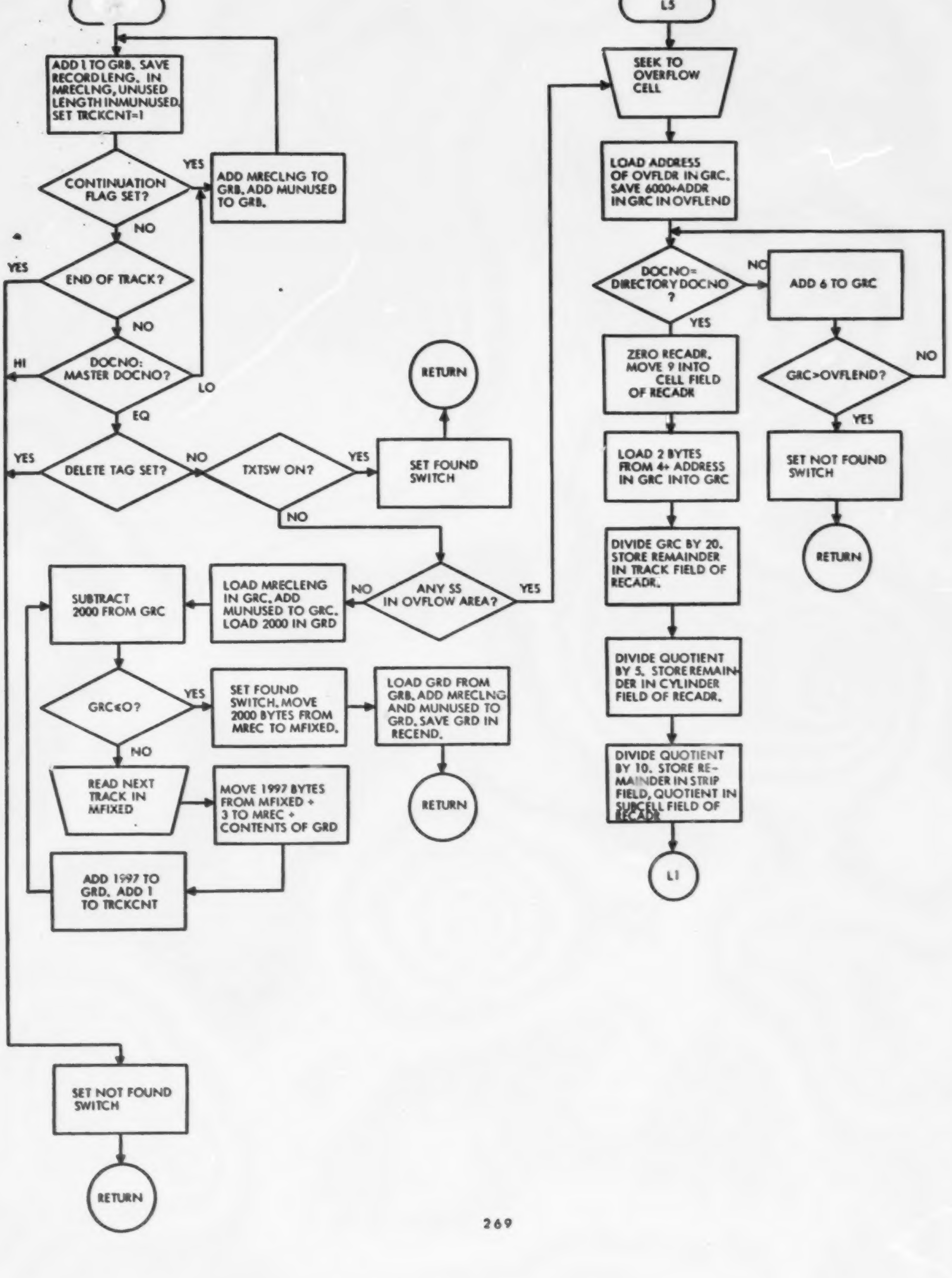

DCELLDMP
OPEN TAPE FILES
LKMF
READ IN DIRECTORIES FROM DATA CELL
SET UP TO READ 1ST RECORD
LKMAST
READ NEXT RECORD
ANY MORE RECORDS
NO
EOJ
YES
SEARCHABLE SEC
NO
YES
IS REC DELETED ?
YES
LKMF
GET TEXT SEGMENT
ANY MORE
NO
YES
NO
BLDRTBL
BUILD RANK TABLE
BLKCHNG
DO BLOCK CHANGES
ENDDOC
WRITE REC ON TAPE


BLKCHNG
ANY BLOCK CHANGES ?
NO
RETURN
YES
SAVE REGISTERS. POSITION TO POINTER TABLE
POSITION TO NEXT POINTER
END OF POINTER TABLE ?
YES
RETURN
NO
POSITION TO DESCRIPTOR. TURN SBDSCRSW OFF
BSEARCH
SEARCH BLOCK CHANGE TABLE FOR DESCRIPTOR
POSITION TO NEXT POINTER
YES
LAST SUBDESCRIPTOR ?
NO
FOUND?
NO
POSITION TO SUBDESCRIPTOR TURN SBDSCRSW ON.
BSEARCH
SEARCH TABLE FOR SUBDESCRIPTOR
YES
FOUND?
YES
ADD 1 TO COUNT FIELD IN TABLE ENTRY
DELETE SUBDESCRIPTOR
YES
TABLE CODE= DELETE?
NO
YES
POSITION TO REPLACING DESCR IN TABLE
SBDSCRSW ON?
NO
SITION PAST ALUES
NO
ID-FILE DESCRIPTOR?
NO
YES
MOVE DESCRIPTOR/SUBDESCRIPTOR INTO MASTER RECORD
YES
TABLE ENTRY LENGTH= MASTER LENGTH
NO
TURN IDSW ON
DIA
DELETE DESCRIPTOR
TURN IDSW ON
NO
LAST SUBDESCRIPTOR ?
YES
SBDSCRSW ON?
SUBTRACT TABLE LENGTH FROM MASTER LENGTH
YES
ID-FILE DESCRIPTOR?
RESULT NEGATIVE?
YES
COMPLEMENT RESULT AND STORE IN SHIFTLNG
SHIFTL
SHIFT RECORD LEFT SHIFTLNG BYTES
STORE RESULT IN SHIFTLNG
SHIFTR
SHIFT RECORD RIGHT SHIFTLNG BYTES
MOVE DESCRIPTOR/SUBDESCRIPTOR INTO MASTER RECORD
TEST FUNCTION BYTE OF REPLACING DESCRIPTOR IN TABLE
YES
SBDSCRSW ON?
NO
BLDINV
CREATE INPUT FOR INVERTED-FILE UPDATE


BSEARCH
SAVE REGISTERS. MOVE NE TO NEREM. LOAD ADDRESS OF TABLE IN GRA
LOAD GRB WITH (LE*NEREM). ADD GRA TO GRB.
LOAD GRC FROM NEREM. DIVIDE GRC BY 2. IGNORE REMAINDER.
LOAD GRD FROM NEREM. SUBTRACT GRC FROM GRD. STORE GRD IN NEREM
LOAD GRB FROM GRC
MULTIPLY GRC BY LE. ADD GRA TO GRC.
NO
NO
GRC≤GRA
LO
ARGUMENT: TABLE ENTRY
HI
GRC≥GRB
YES
EQ
YES
TURN FOUNDSW OFF.
TURN FOUNDSW ON.
TURN FOUNDSW OFF.
RETURN


EOJ
ANY BLOCKCHANGES?
NO
RETURN
YES
POSITION GRC TO TABLE OF BLOCK CHANGES
COUNT FIELD= ZERO
YES
PRINT MESSAGE: DESCRIPTOR NOT APPLIED IN BLOCK CHANGES.
NO
SET TYPE= DELETE ALL. SET DOCNO=O
INCREMENT GRC BY LENGTH OF TABLE ENTRY
END OF TABLE
NO
YES
BLDINV
CREATE INPUT TO INVERTED-FILE UPDATE
RETURN


LOOKUP
SAVE REGISTERS ZERO RECADR, LOAD ADDRESS OF MAST BUF IN GRB
FIRST TIME SWITCH ON?
YES
TURN FIRST TIME SWITCH OFF. SET RECADR=CELLO, SUBCELLO STRIPO, CYLINDER O, TRACK 1
SEEK TO RECADR
READ TRACK 1, 2, 3, 4 INTO STRPDR
PUT STRPDR ON FOUR TRACKS OF DISK
READ TRACK 5, 6, 7 INTO OVFLDR
PUT OVFLDR ON 3 TRACKS OF DISK
NO
LOAD ADDRESS OF STRPDR IN GRC. ADDRESS OF CELLDR IN GRD.
END OF CELL DR?
YES
SET NOT FOUND SWITCH
RETURN
NO
DOCNO≤DIRECTORY DOCNO
NO
ADD 800 TO GRC, ADD4 TO GRD
YES
SAVE 800+ADDR IN GRC IN STRPEND
GRC>STRPEND?
YES
NO
DOCNO≤DIRECTORY DOCNO
NO
ADD LENGTH OF STRIP DIRECTORY ENTRY TO GRC. SAVE GRC IN SADR.
YES
SUBTRACT ADDR OF STRPDR FROM GRC. DIVIDE GRC BY LENGTH OF STRIP DIRECTORY ENTRY
DIVIDE QUOTIENT BY 10. STORE REMAINDER IN STRIP FIELD OF RECORD ADDR.
DIVIDE QUOTIENT BY 20. STORE REMAINDER IN SUBCELL FIELD OF RECORD ADDR.
L2
L2
DIVIDE QUOTIENT BY 10 STORE REMAINDER IN CELL FIELD OF RECORD ADDR
SEEK TO RECADR
READ TRACK O INTO TRCKDR
LOAD ADDRESS OF TRCKDR IN GRC.
DOCNO≤DIRECTORY DOCNO
YES
NO
ADD 4 TO GRC
SUBTRACT ADDR OF TRCKDR FROM GRC. DIVIDE GRC BY 4.
DIVIDE QUOTIENT BY 20. STORE QUOTIENT IN CYLINDER FIELD, REMAINDER IN TRACK FIELD OF ADDR
SEEK REQUIRED?
YES
SEEK TO RECADR
NO
READ 1 TRACK FROM RECADR INTO MREC
L4


ADD 1 TO GRB. SAVE RECORD LENG. IN MRECLNG, UNUSED LENGTH IN MUNUSED, SET TRCKCNT=1
CONTINUATION FLAG SET?
YES
ADD MRECLNG TO GRB. ADD MUNUSED TO GRB.
NO
YES
END OF TRACK?
NO
HI
DOCNO: MASTER DOCNO?
LO
EQ
YES
DELETE TAG SET?
NO
TXTSW ON?
YES
SET FOUND SWITCH
RETURN
NO
ANY SS IN OVFLOW AREA?
NO
YES
LOAD MRECLENG IN GRC. ADD MUNUSED TO GRC. LOAD 2000 IN GRD
SUBTRACT 2000 FROM GRC
GRC≤O?
YES
SET FOUND SWITCH. MOVE 2000 BYTES FROM MREC TO MFIXED.
LOAD GRD FROM GRB. ADD MRECLNG AND MUNUSED TO GRD. SAVE GRD IN RECEND.
RETURN
NO
READ NEXT TRACK IN MFIXED
MOVE 1997 BYTES FROM MFIXED + 3 TO MREC + CONTENTS OF GRD
ADD 1997 TO GRD. ADD 1 TO TRCKCNT
SET NOT FOUND SWITCH
RETURN
L5
SEEK TO OVERFLOW CELL
LOAD ADDRESS OF OVFLDR IN GRC. SAVE 6000+ADDR IN GRC IN OVFLEND
DOCNO= DIRECTORY DOCNO ?
NO
ADD 6 TO GRC
YES
GRC>OVFLEND?
NO
YES
ZERO RECADR. MOVE 9 INTO CELL FIELD OF RECADR
SET NOT FOUND SWITCH
RETURN
LOAD 2 BYTES FROM 4+ ADDRESS IN GRC INTO GRC
DIVIDE GRC BY 20. STORE REMAINDER IN TRACK FIELD OF RECADR.
DIVIDE QUOTIENT BY 5. STORE REMAINDER IN CYLINDER FIELD OF RECADR.
DIVIDE QUOTIENT BY 10. STORE REMAINDER IN STRIP FIELD, QUOTIENT IN SUBCELL FIELD OF RECADR
L1

DATA CELL RELOAD PROGRAM (LOADCELL) PROCESSING LOGIC

LOADCELL reads records from the Master file tape and writes them onto the data cell creating and updating directories to the Master file. The records are printed using MPRINT1 which acts as a subroutine to LOADCELL. When an end of file is read from the tape, the directories and control information such as the highest internal number, next available address on the data cell,etc. are written on the data cell and the program terminates.

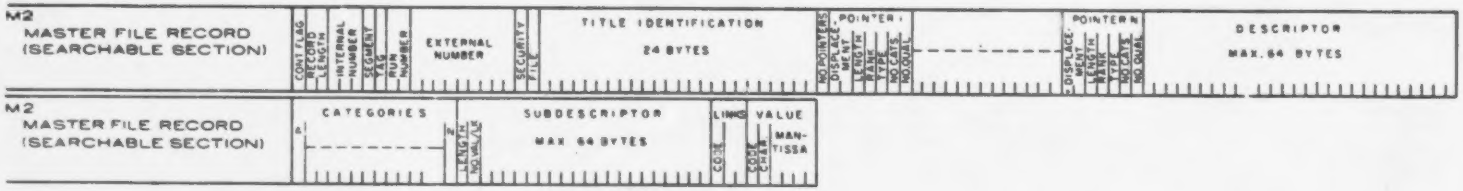
Input

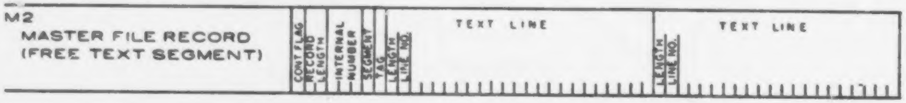
The tape (format M2 below) created by DUMPCELL provides input to LOADCELL.

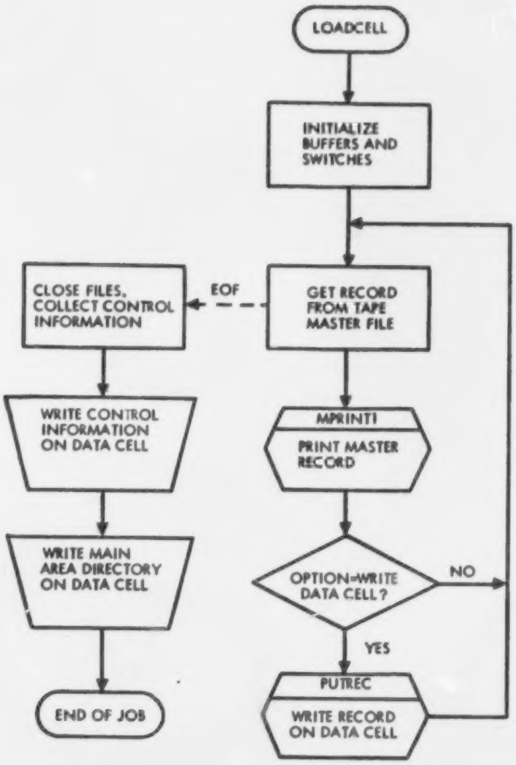
Output

Records in the M2 format below are written onto the data cell.

DATA CELL RELOAD PROGRAM (LOADCELL) RECORD FORMATS


M2
MASTER FILE RECORD
(SEARCHABLE SECTION)
CONT FLAG
RECORD LENGTH
INTERNAL NUMBER
SEGMENT
TAG
RUN NUMBER
EXTERNAL NUMBER
SECURITY
FILE
TITLE IDENTIFICATION
24 BYTES
NO POINTERS
POINTER 1
DISPLACE-MENT
LENGTH
RANK
TYPE
NO CATS.
NO QUAL.
POINTER N
DISPLACE-MENT
LENGTH
RANK
TYPE
NO CATS
NO QUAL
DESCRIPTOR
MAX. 64 BYTES
M2
MASTER FILE RECORD
(SEARCHABLE SECTION)
CATEGORIES
1
N
LENGTH
NO VALUE
SUBDESCRIPTOR
MAX 64 BYTES
LING
CODE
VALUE
CODE
CHAR
MAN-TISSA


M2
MASTER FILE RECORD
(FREE TEXT SEGMENT)
CONT FLAG
RECORD LENGTH
INTERNAL NUMBER
SEGMENT
TAG
LENGTH
LINE NO.
TEXT LINE
LENGTH
LINE NO.
TEXT LINE


LOADCELL
INITIALIZE BUFFERS AND SWITCHES
GET RECORD FROM TAPE MASTER FILE
EOF
CLOSE FILES, COLLECT CONTROL INFORMATION
WRITE CONTROL INFORMATION ON DATA CELL
WRITE MAIN AREA DIRECTORY ON DATA CELL
END OF JOB
MPRINT1
PRINT MASTER RECORD
OPTION=WRITE DATA CELL ?
NO
YES
PUTREC
WRITE RECORD ON DATA CELL

SEARCHING SYSTEM

Search Processing Logic

The basic structure of the FDA search program is a core resident root phase and three overlay phases residing on disk and loaded into core storage for execution. Information is passed between the three phases by a common area in the root phase. Phase 1 reads the request input and builds a request string in common. Phase 2 uses the inverted file on disk to build a list of likely documents to match the request being processed. Phase 3 scans each document in this list and outputs each hit.

Root Phase

Within the root phase is a small calling program which loads phase 1 into core and branches to it. If there are no errors in this request, then phase 2 and phase 3 are called depending upon the options requested. If phase 1 detects any errors, then the errors are noted on the printer and no further processing is done with this request. After each request has been processed, phase 1 is loaded again and the above procedure is repeated until all requests have been processed. Also within this calling program in the root phase are a few general subroutines which are used by all three phases. XMOVE is a generalized move subroutine where the general registers RW, RX, RY, and RT are the from address, the to address, the length of data to move, and the return address respectively. PUTRM is a subroutine which moves a term from the QSECT area of common to the request string while GETRM gets the next term in the request string and moves it to QSECT. These subroutines are similar to the GET and PUT with WORKA = YES in logical IOCS. XPAGE is a subroutine which prints the title and page number at the top of each page of printer output.

Common and I/O Areas

Also within the root phase is a common area which contains branch addresses, counts, constants, and work areas which are used by all three phases. TABL1, WORK2, and WORK3 are for the table of file pointers (see phase 2), the request string and a disk input/output area. Phase 1 uses WORK1 for the dictionary records and WORK6A and WORK6B for the tables of files and context codes. These are overlaid during phase 2 by WORK5A, WORK5B, and WORK5C which are the three document number string buffers. These are in turn overlaid by WORK4 and WORK4G during phase 3. WORK4 contains the master file document record and WORK4G is the RPG tape output area.

DADIO .

The direct access device input/output routine performs all reading and writing of the disk and data cell. All linkage to DADIO is generated by the macros RDSK, WDSK, RDCL, and WDCL for read (R) or write (W) disk (DSK) or data cell (DCL).

PLINE

The print/type routine performs all input/output with the console typewriter and the printer. Linkage is generated by the PRNT and TYPE macros. This program also keeps a line count and will execute the users page heading routine when a page is filled.

Phase 1 (edit request input)

This phase edits the input request and builds the request string in common. The request string consists of variable length records with each record being either a descriptor, a subdescriptor, a truncated descriptor, one or two values, a logical operator, or an indicator to start a scan or link operation. Additional information is kept in common for each request such as the title line, the kind of search request, and the kind of output wanted.

SEDIT

This routine edits the input request card set. The TITLE, SEARCH PRINT, REPORT, and SELECT cards (the latter two are for RPG output only) are processed and switches are set in common depending upon the options entered. Next, the REQUEST cards are read and processed. If there is both a document number string and a Boolean expression in the request, the document numbers must precede the Boolean expression. This condition exists when SEARCH TYPE = MIXED. If there is a document number string, it is output in the same format as described in phase 2 for the inverted file document number strings. Each descriptor and subdescriptor in the input request is verified with the dictionary file. Each value is converted to a floating point number format.

Special Characters

There are eight special characters in the input search language. They serve as logical operators and separators. They are defined as follows:

SYMBOLIC	CHARACTER	HEXADECIMAL	FUNCTION
EQAND	&	X'50'	Logical AND operator
EQLOR	|	X'4F'	Logical OR operator
EQNOT	¬	X'5F'	Logical NOT operator
EQCMA	;	X'5E'	Comma separator
EQPER	:	X'7A'	Period separator
EQLPA	<	X'6E'	Left parenthesis
EQERQ	>	X'6F'	Right parenthesis

The input request is scanned character by character until one of these special characters is encountered. Multiple blanks are reduced to one blank and leading and trailing blanks are discarded on the data between any two of these special characters. This data is processed as follows:

CURRENT CHARACTER (DATA LENGTH IS ZERO)

LAST	OPERATOR	;	:	<	>	?
OPERATOR	(NO)	(NO)	(NO)	VALID	(NO)	(NO)
;	(NO)	(NO)	(NO)	(NO)	(NO)	(NO)
:	(NOTE3	NOTE3	(NO)	(NO)	NOTE3	NOTE3
<	(NO)	(NO)	(NO)	VALID	(NO)	(NO)
>	VALID	(NO)	(NO)	(NO)	VALID	VALID
(NONE)	(NO)	(NO)	(NO)	VALID	(NO)	(NO)

CURRENT CHARACTER (DATA LENGTH NOT ZERO)

LAST	OPERATOR	;	:	<	>	?
OPERATOR	DESC	DESC	(NO)	NOTE2	DESC	DESC
;	SUBD	SUBD	NOTE1	(NO)	SUBD	SUBD
:	VALUE	VALUE	(NO)	(NO)	VALUE	VALUE
<	DESC	DESC	(NO)	(NO)	DESC	(NO)
>	(NO)	(NO)	(NO)	(NO)	(NO)	(NO)
(NONE)	DESC	DESC	(NO)	YOTE3	(NO)	DESC

(NO)	indicates that this sequence is invalid
NOTE1	must be EQ, NE, GT, GE, LT, or LE followed by a value
NOTE2	FM, FC, EN, or EC for scan or L for links
NOTE3	a value was processed in CVFPN program

Numeric Operators

There are six numeric operators mentioned in NOTE1 above. One of these six must precede each value in the input request. Each numeric operator is converted to a numeric indicator in the request string according to the following table:

		V>S	V=S				V<S
OPERATOR	INDICATOR	V>U>L		V=U>L	U>V>L	U>V=L	U>L>V
EQ:	X'1E'		HIT	HIT	HIT	HIT	
NE:	X'21'	HIT					HIT
GT:	X'07'				HIT	HIT	HIT
GE:	X'1F'		HIT	HIT	HIT	HIT	HIT
LT:	X'2C'	HIT		HIT			
LE:	X'3E'	HIT	HIT	HIT	HIT	HIT	
		X'20'	X'10'	X'08'	X'04'	X'02'	X'01'

V	is input request value
S	is a single value in document record
U	is upper value in document record
L	is lower value in document record

Scan or Link

Each descriptor in the document record has a rank order or position. The requestor may specify that descriptors must be ranked within the document in a particular order. The rank scan option generates one record in the request string indicating the start of the scan option and inserts an indicator in each descriptor record affected by this scan. The indicators are X'0A' for FC (forward contiguous, ranks must be in consecutive ascending order), X'02' for FN (forward noncontiguous, ascending but not necessarily consecutive), X'0E' for EC (either way contiguous, consecutive ranks in either ascending or descending order), and X'06' for EN (either way noncontiguous, need not be consecutive but must all be either in ascending or descending order). Descriptors may be linked together and subdescriptors may be linked together. The indicator for linking (L) is X'01' and is inserted into the request string in the same manner as the scan option.

LKDICT

The dictionary look up routine is used to verify each descriptor and subdescriptor in the input request. This is the same program used in the master file edit program (MFEDIT). The dictionary file is an indexed sequential disk file.

CVFPN

The convert to floating point number routine scans and validates all input request numbers and converts each to a floating point number format. Date and time values are also processed by this routine. When entering this routine, register 1 points to the number in the request to be converted. When leaving this routine, register 1 points to the last character processed in the input card,register 0 points to the converted output. The output contains the following:

byte	0	code	0 - value, X'11 - date, X'21' - time, X'C3' - context code
byte	1	length	0 - error, else 1 - 9
byte	2	sign	0 - minus - L - plus
byte	3	exponent	X'80' is zero, X'81' is one, etc.
bytes	4 - 12	characters	up to nine EBDIC characters

XSORT

This is a small, generalized core sort routine. Linkage is generated by the XSRT macro. It is used for sorting document numbers during phase 1

Phase 2 (Inverted File Search

This phase builds the document number string used in phase 3. The request string is scanned for descriptors and logical operators. A string of document numbers is retrieved for each descriptor and two stirings are merged into one string for each logical operator. There is one more descriptor than are logical operators in the request string so that there is one document number string at the completion of this phase of the search program. Due to the potentially large list of document numbers, each document number string is stored on a disk work file.

SIVRT

This routine performs the logical operations on the document number strings retrieved from the inverted file. This routine uses three core buffers of document number strings. The last two strings on the disk work file are moved into two of the buffers and the resultant string is built in the third buffer. This resultant string is the result of the logical AND, OR, or NOT on the other two strings and replaces them on the disk work file.

File Pointers

A table of file pointers is used to point to the document number strings on the disk work file. At least one file pointer is needed for each string and three file pointers can be used to point to a string containing more than 240,000 document numbers occupying about 20 cylinders on the work disk. Each file pointer is 16 bytes in length. There are two bytes each for a disk track address, a displacement within that track, a length within that track, and a two byte work area There is one byte each for track count this string, current count, switch for pointer number and a code for type of document number string (all, none, or mixed). There is also a full word work area in each file pointer which is used to store an address constant during ANDing, ORing, and NOTing operations.

Descriptors and Document Number Strings

At least one file pointer is generated for each descriptor in the request string. A file pointer with type code of all is generated for each common descriptor. Each precise descriptor is looked up in the inverted file and its document number string is moved to the disk work file and a file pointer is generated. If a precise descriptor has no string in the inverted file, a file pointer with a type code of none is generated. Since there can be eight levels of nested parentheses, there may be as many as nine document number strings on the disk work file at one time.

Logical Operators and Operations

When a logical operator is encountered in the request string, the last two document number strings on the disk work file are logically operated upon to form the resultant string. The table below describes the resultant document number string (or file pointer) after logically operating on strings A and B. STRING indicates that a document number string exists for this descriptor while ALL indicates a common descriptor and NON means this precise descriptor has no document number in the inverted file. The last three columns in the table give the resultant string and file pointer after the logical operation. To logically AND two strings of document numbers, the document numbers in strings A and B are individually compared. Only when the same document number exists in both strings is it put in the resultant string. The logical OR operation on strings A and B forms a resultant string of all document numbers from string A plus any additional document numbers in string B. The resultant string of a logical NOT operation contains all document numbers from string A except that those in string B which compare equal to those in string A are rejected.

A	B	LOGICAL AND	LOGICAL OR	LOGICAL NOT
STRING	STRING	A & B	A | B	A ¬ B
STRING	NONE	NONE	A	A
STRING	ALL	A	ALL	A
ALL	STRING	B	ALL	ALL
ALL	NONE	NONE	ALL	ALL
ALL	ALL	ALL	ALL	ALL
NONE	STRING	NONE	B	NONE
NONE	NONE	NONE	NONE	NONE
NONE	ALL	NONE	ALL	NONE

As the request string is formed during the edit phase of the search, the logical operators are reordered so that they follow the descriptors which they reference. For example, the request A & B becomes A B &. This allows sequential processing through the request string. This allows retrieval of document numbers for A and B in the inverted file and then performing the logical operation on these two strings. This method of ordering logical operators greatly simplifies the processing of a request especially when several levels of nested parentheses are used.

SDOUT

The document number output routine prints the document numbers in the document number string on the disk work file. This routine is called after the SIVRT routine has completed processing and either an inverted file search is requested or after the output limit has been exceeded during phase 3 or an ID. FILE search.

LKIVRT

The inverted file look up routine finds each requested precise descriptor in the inverted file and passes its document number string to the SIVRT routine. This file is a disk resident indexed sequential file maintained on magnetic tape.

Phase 3 (Master File Search)

This last phase of the search program scans each document in the document number string on the disk work file and outputs each list. An ID. file search scans each document in the identification file tape and there is no document number string for this type of search. However, an ID. file search outputs a document number string instead of outputting each hit. A document search contains no Boolean expression, therefore, each document in the string is output and not scanned.

SFSEQ

This routine is the master file sequential search routine. It retrieves a document from the master file and compares it with the request string. This routine compares subdescriptors and values as well as the checking of common descriptors, linking, and descriptor rank or position in the document.

SMOUT

This routine outputs the master file record from the master file record buffer. Output is on the printer or on magnetic tape or both and consists of the searchable section and/or text segments.

LKMAST

The master file look up routine retrieves the requested document in the master file if it is there and also retrieves text segments.

Input

Input to the search system will be from request cards, magnetic tape, disk files, and the 2321 data cell. The card formats are described in detail in the search language section of the User's Manual.

1. Tape

   The ID. file on magnetic tape will be in the format labelled M2 below. No free text is included. For a detailed description of the searchable section of the record phase, refer to Appendix B.

2. Disk

   a. The dictionary records will be in the format labelled D2 below. A detailed description of the dictionary master record is contained in Appendix A.

b. The inverted file records are in the format (M5) below. For a detailed layout of the inverted file records, please refer to Appendix C. Note that only the descriptor and its associated document numbers are on the disk.

c. The cross reference file records are in the format labelled M4 below.

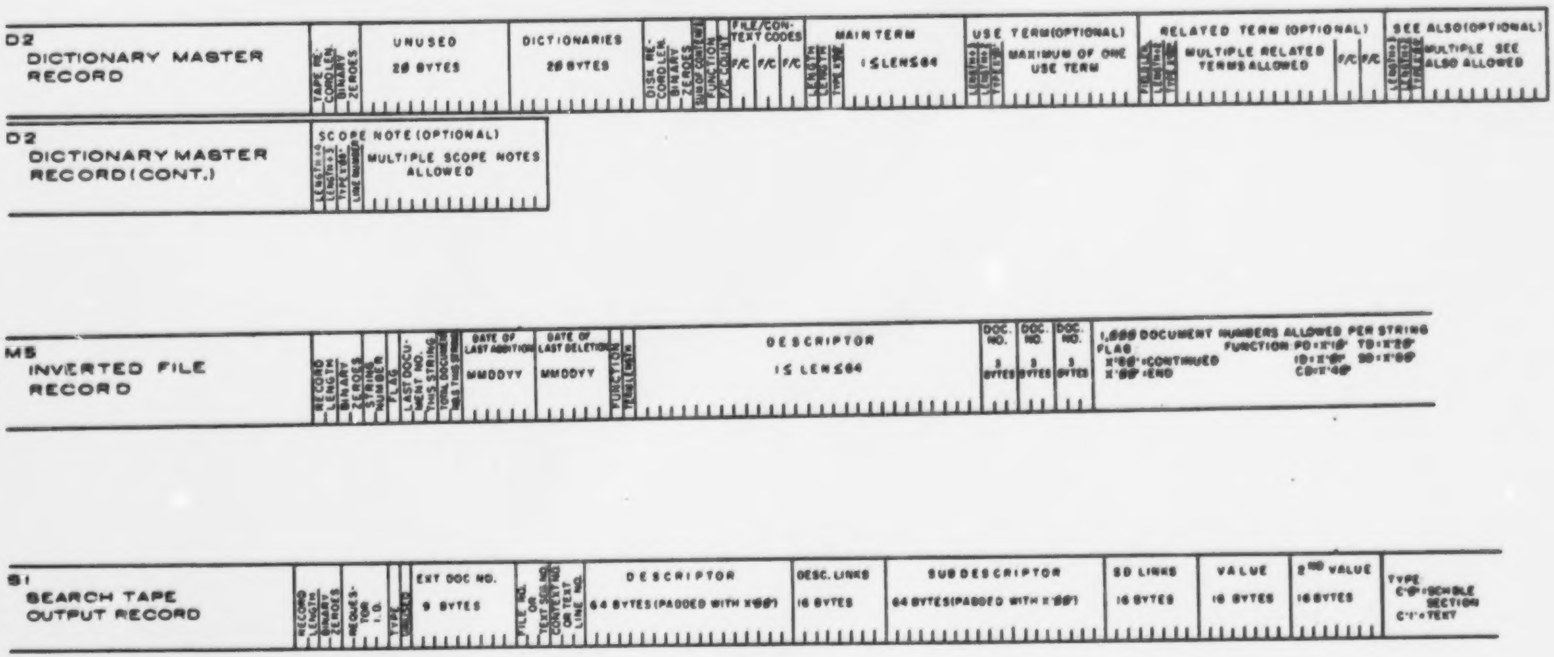
3. Data Cell

The master file resides on the 2321 data cell. The format of the master file records is shown below labelled M2. For a detailed description of the master file record format, please refer to Appendix B.

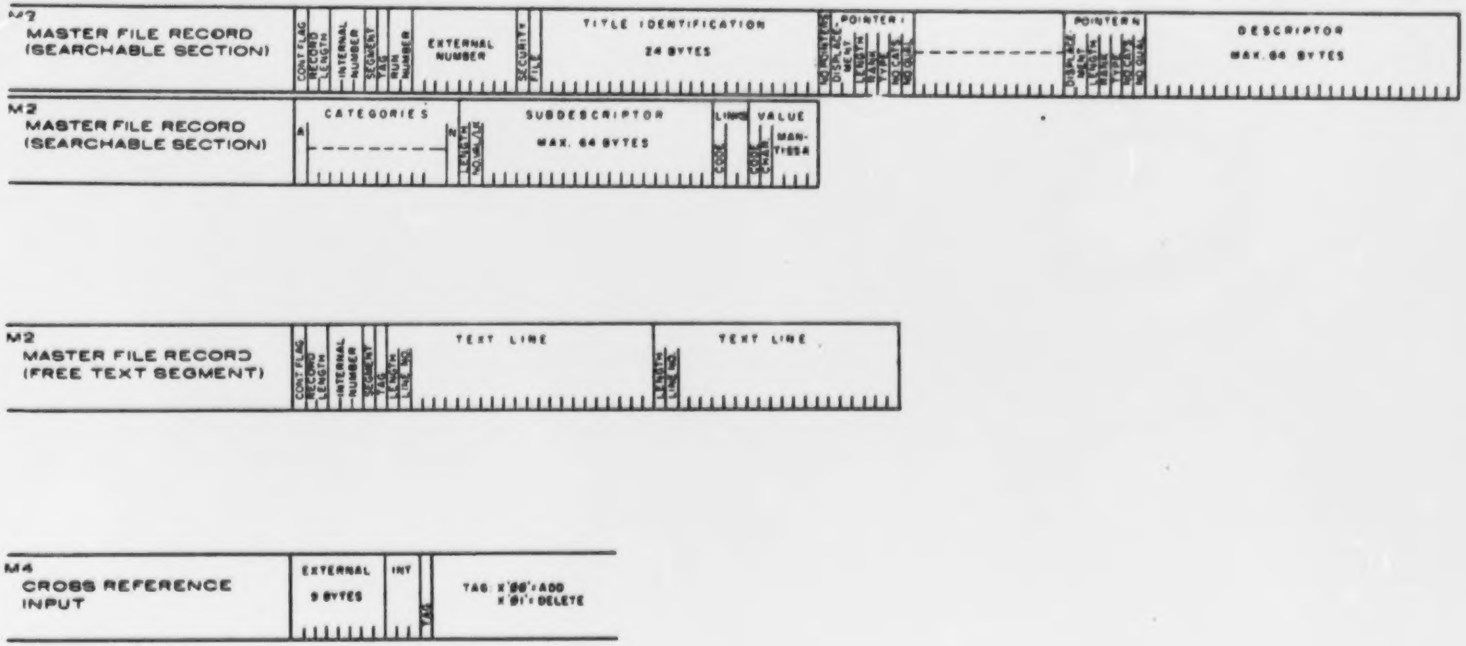
Output

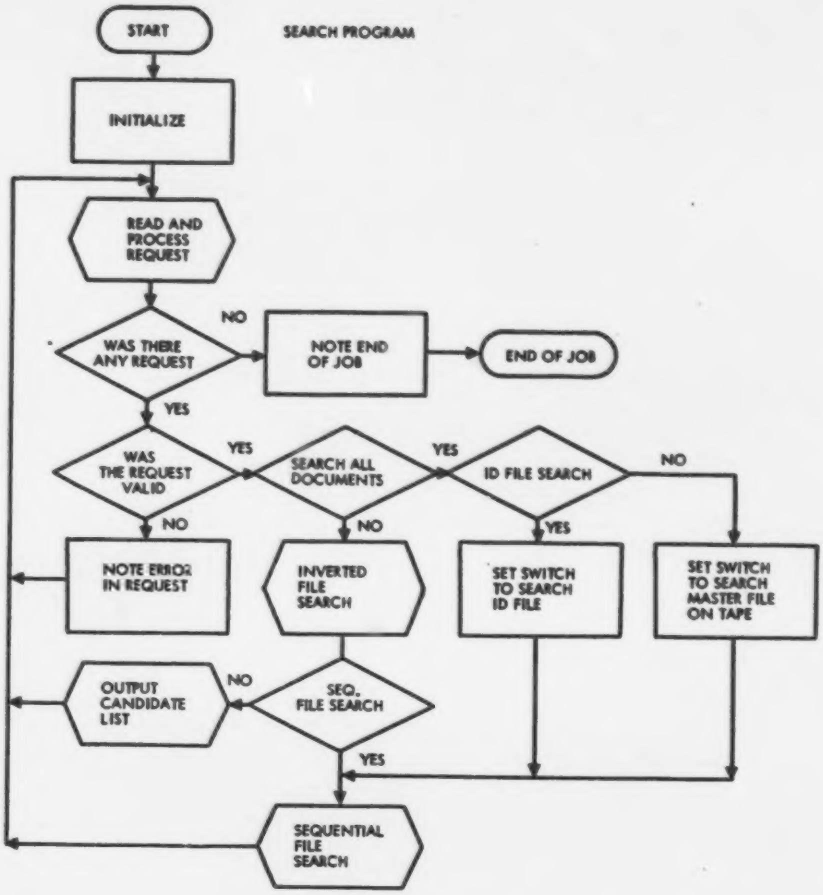
The output from the search may be in either or both of two forms: a printed listing or a magnetic tape suitable for further processing. The format of the tape record is shown below labelled S1. For a detailed description of the search output record, please refer to Appendix E.

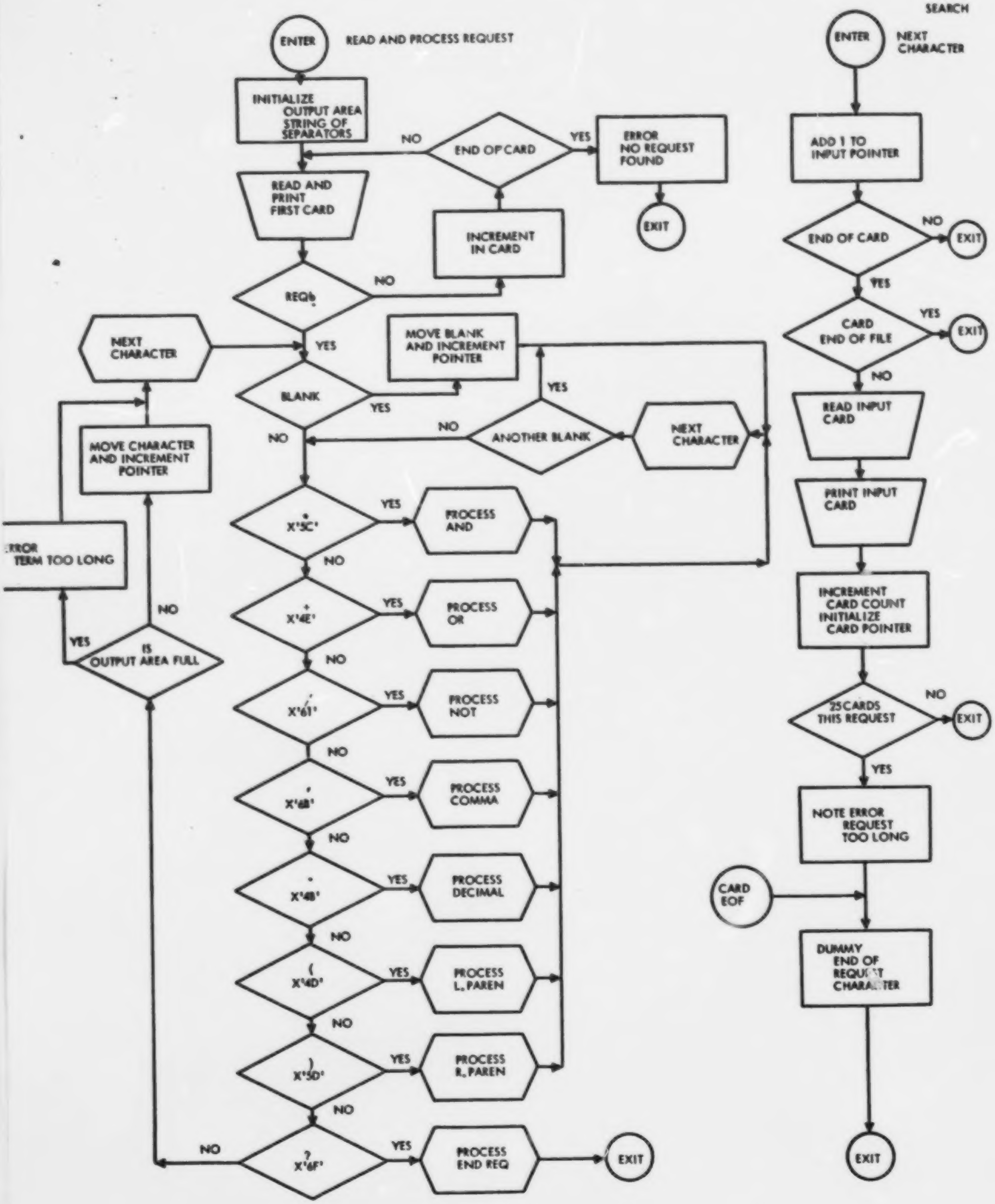
SEARCHING SYSTEM RECORD FORMATS

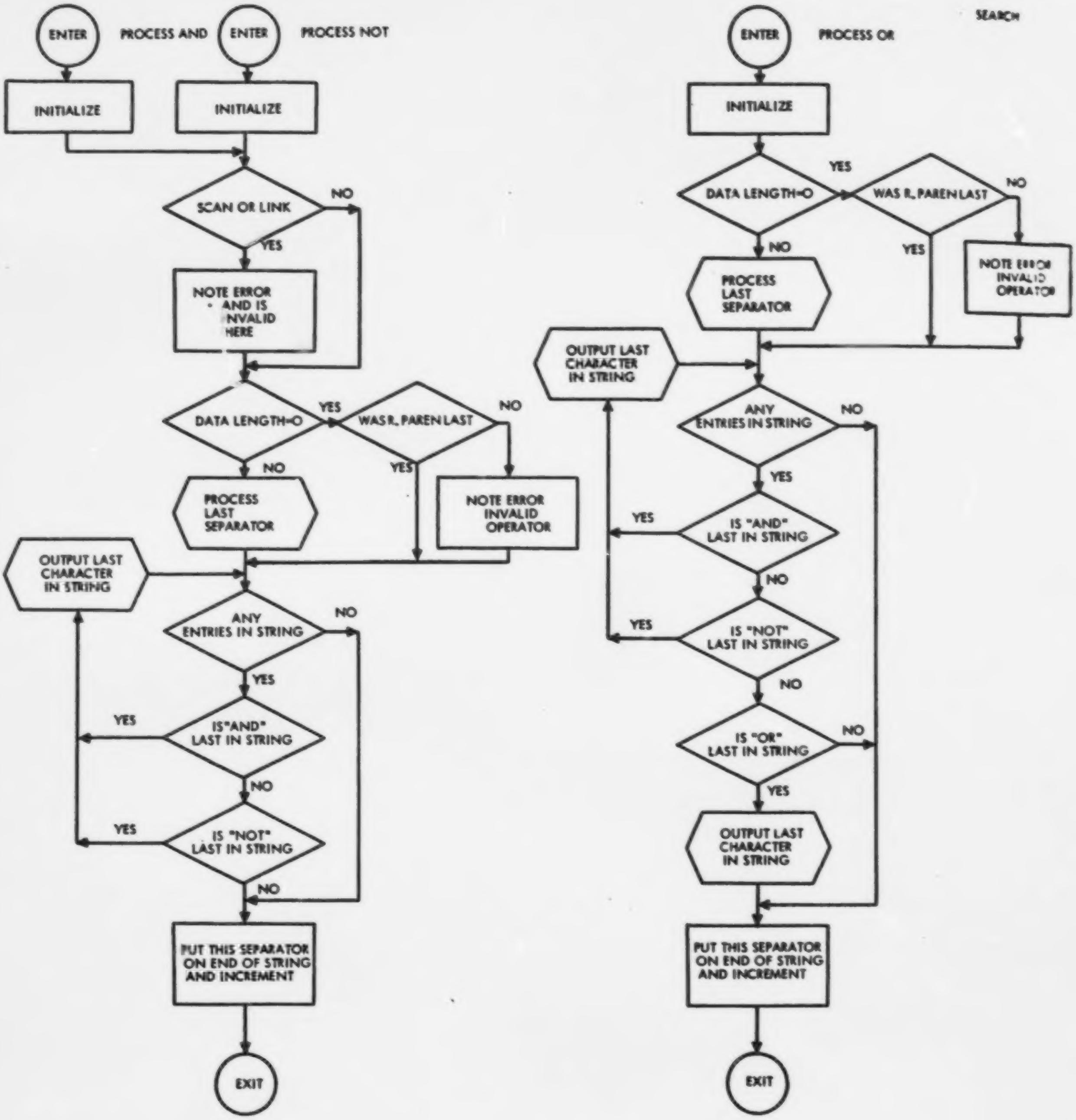

D2
DICTIONARY MASTER RECORD
TAPE RE-CORD LEN.
BINARY ZEROES
UNUSED
20 BYTES
DICTIONARIES
20 BYTES
DISK RE-CORD LEN.
BINARY ZEROES
FUNCTION
F/C COUNT
FILE/CON-TEXT CODES
F/C
F/C
F/C
MAIN TERM
LENGTH
LENGTH
1≤LEN≤64
USE TERM(OPTIONAL)
MAXIMUM OF ONE USE TERM
RELATED TERM (OPTIONAL)
MULTIPLE RELATED TERMS ALLOWED
F/C
F/C
SEE ALSO(OPTIONAL)
MULTIPLE SEE ALSO ALLOWED
D2
DICTIONARY MASTER RECORD(CONT.)
SCOPE NOTE (OPTIONAL)
LINE NUMBER
MULTIPLE SCOPE NOTES ALLOWED
M5
INVERTED FILE RECORD
RECORD LENGTH
BINARY ZEROES
STRING NUMBER
FLAG
LAST DOCU-MENT NO. THIS STRING
TOTAL DOCUMENTS THIS STRING
DATE OF LAST ADDITION
MMDDYY
DATE OF LAST DELETION
MMDDYY
FUNCTION
TERM LENGTH
DESCRIPTOR
1≤LEN≤64
DOC. NO.
3 BYTES
DOC. NO.
3 BYTES
DOC. NO.
3 BYTES
1,000 DOCUMENT NUMBERS ALLOWED PER STRING
FLAG
X'00'=CONTINUED
X'80'=END
FUNCTION: PO=X'10' TO=X'20'
ID=X'8F' SD=X'80'
CD=X'40'
S1
SEARCH TAPE OUTPUT RECORD
RECORD LENGTH
BINARY ZEROES
REQUES-TOR I.D.
TYPE
UNUSED
EXT DOC NO.
9 BYTES
FILE NO. OR TEXT SEG.NO. CONTEXT NO. OR TEXT LINE NO.
DESCRIPTOR
64 BYTES(PADDED WITH X'00')
DESC. LINKS
16 BYTES
SUBDESCRIPTOR
64 BYTES(PADDED WITH X'00')
SD LINKS
16 BYTES
VALUE
16 BYTES
2ND VALUE
16 BYTES
TYPE
C'S'=SCHEDULE SECTION
C'T'=TEXT

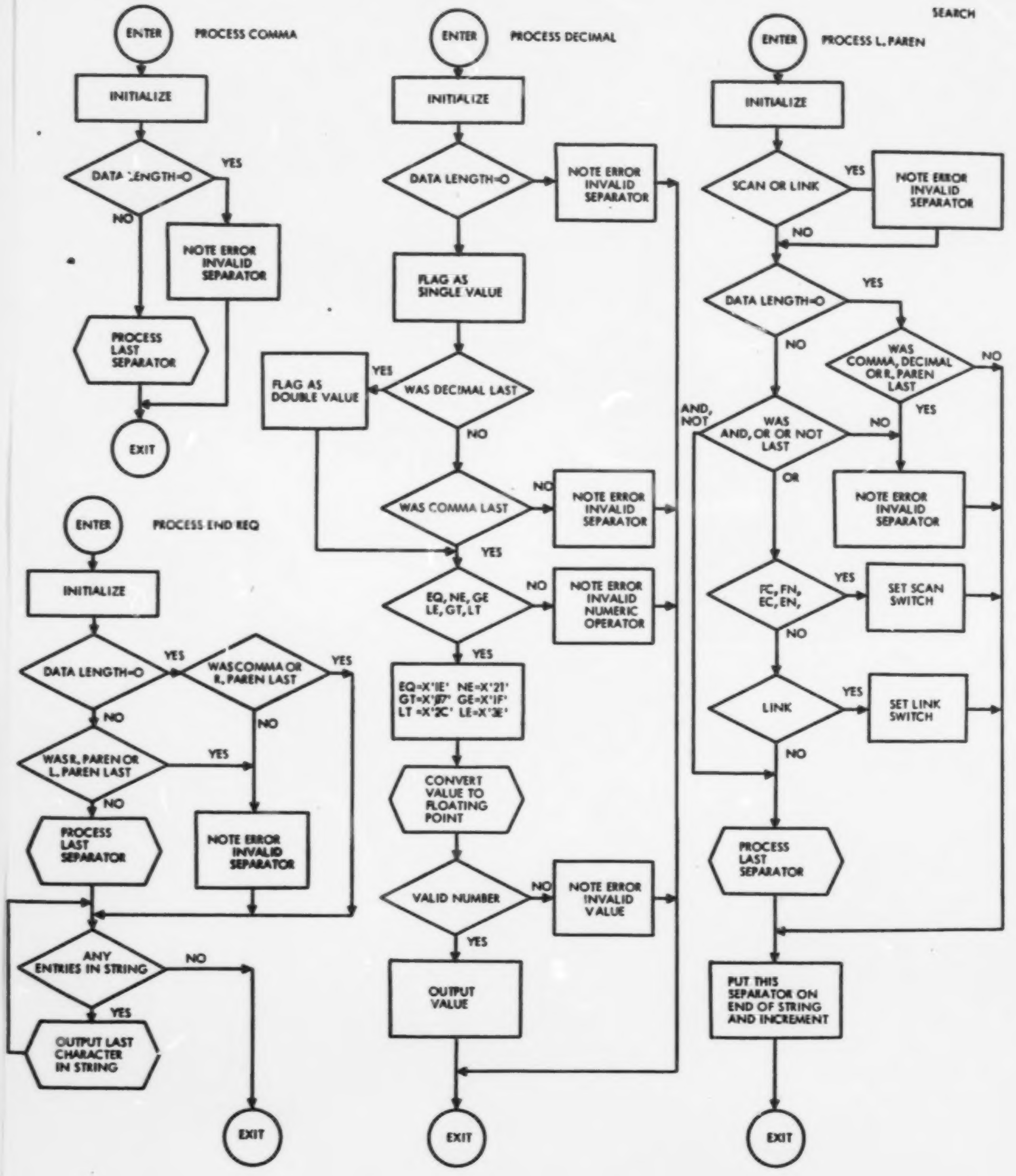
SEARCHING SYSTEM RECORD FORMATS (continued)

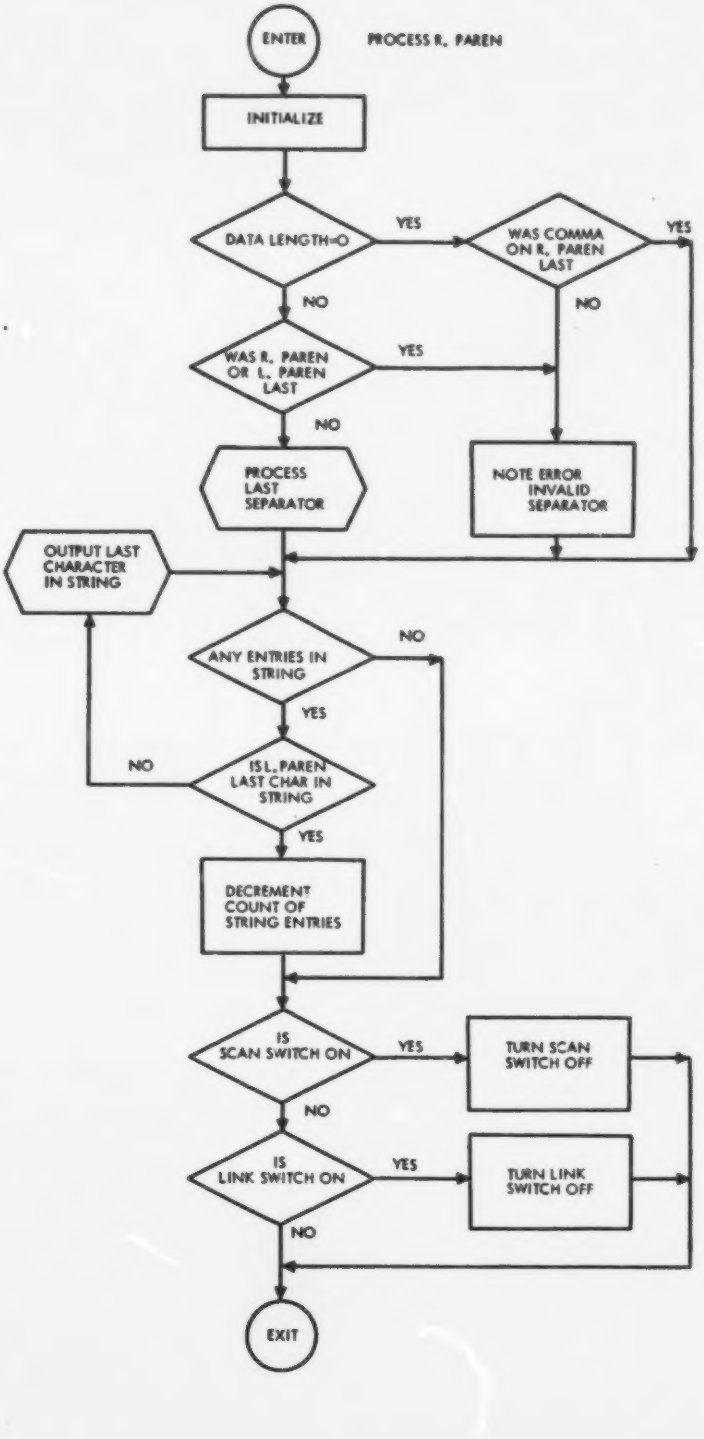

M2
MASTER FILE RECORD (SEARCHABLE SECTION)
CONT FLAG
RECORD LENGTH
INTERNAL NUMBER
SEGMENT
TAG
RUN NUMBER
EXTERNAL NUMBER
SECURITY
FILE
TITLE IDENTIFICATION
24 BYTES
NO POINTERS
POINTER 1
DISPLACE-MENT
LENGTH
RANK
TYPE
NO CATS
NO QUAL
POINTER N
DISPLACE-MENT
LENGTH
RANK
TYPE
NO CATS
NO QUAL
DESCRIPTOR
MAX. 64 BYTES
M2
MASTER FILE RECORD (SEARCHABLE SECTION)
CATEGORIES
A
N
LENGTH
NO VALUE
SUBDESCRIPTOR
MAX. 64 BYTES
LINK
CODE
VALUE
CODE
CHAR
MAN-TISSA
M2
MASTER FILE RECORD (FREE TEXT SEGMENT)
CONT FLAG
RECORD LENGTH
INTERNAL NUMBER
SEGMENT
TAG
LENGTH
LINE NO
TEXT LINE
LENGTH
LINE NO
TEXT LINE
M4
CROSS REFERENCE INPUT
EXTERNAL
9 BYTES
INT
TAG
TAG: X'00': ADD
X'01': DELETE

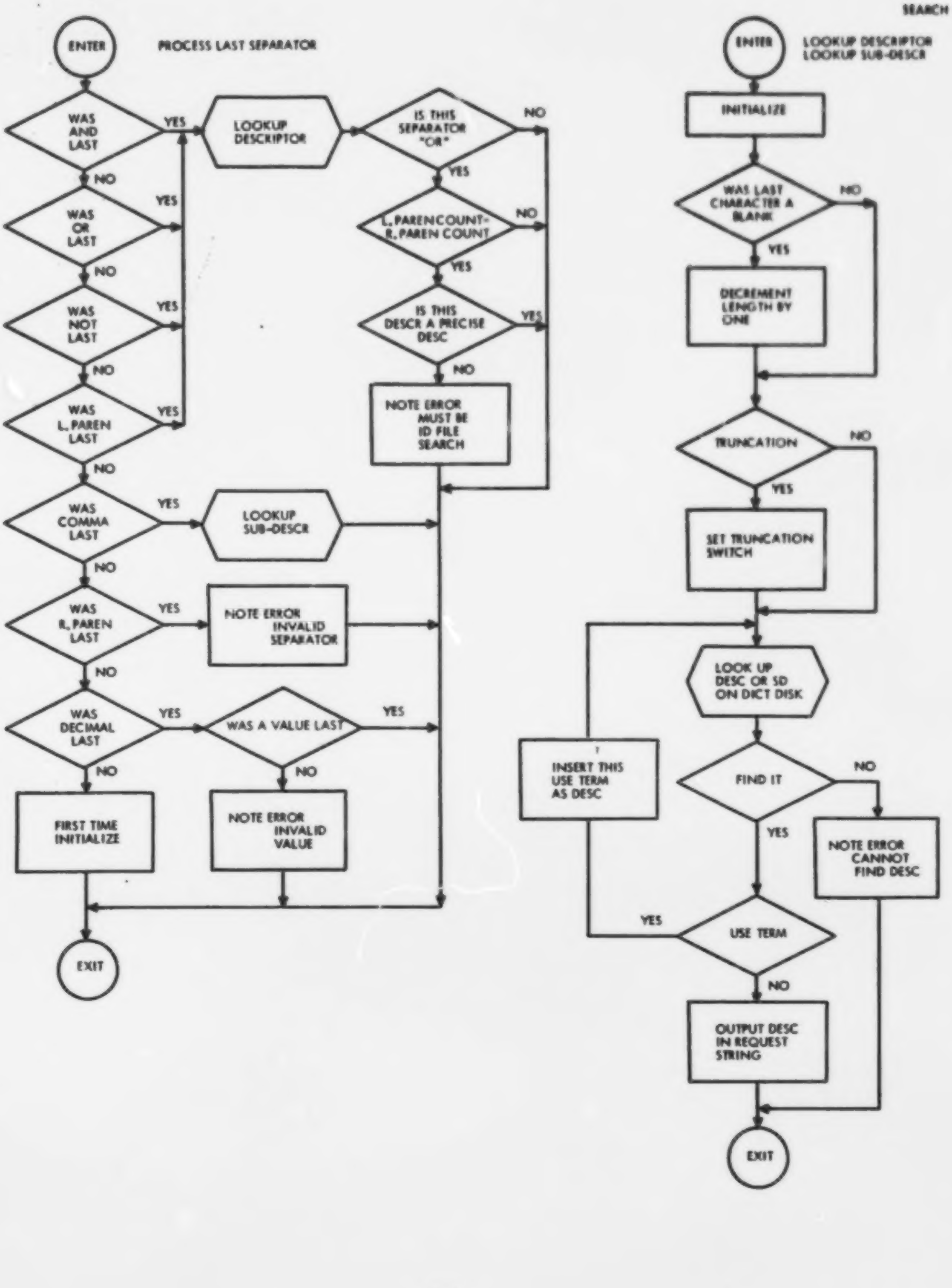

START
SEARCH PROGRAM
INITIALIZE
READ AND PROCESS REQUEST
WAS THERE ANY REQUEST
NO
NOTE END OF JOB
END OF JOB
YES
WAS THE REQUEST VALID
YES
SEARCH ALL DOCUMENTS
YES
ID FILE SEARCH
NO
NO
NOTE ERROR IN REQUEST
NO
INVERTED FILE SEARCH
YES
SET SWITCH TO SEARCH ID FILE
SET SWITCH TO SEARCH MASTER FILE ON TAPE
OUTPUT CANDIDATE LIST
NO
SEQ. FILE SEARCH
YES
SEQUENTIAL FILE SEARCH


SEARCH
ENTER
READ AND PROCESS REQUEST
INITIALIZE OUTPUT AREA STRING OF SEPARATORS
NO
END OF CARD
YES
ERROR NO REQUEST FOUND
EXIT
READ AND PRINT FIRST CARD
INCREMENT IN CARD
REQb
NO
YES
NEXT CHARACTER
MOVE BLANK AND INCREMENT POINTER
BLANK
YES
YES
NO
ANOTHER BLANK
NEXT CHARACTER
NO
MOVE CHARACTER AND INCREMENT POINTER
*
X'5C'
YES
PROCESS AND
NO
RROR TERM TOO LONG
+
X'4E'
YES
PROCESS OR
NO
NO
YES
IS OUTPUT AREA FULL
/
X'61'
YES
PROCESS NOT
NO
,
X'6B'
YES
PROCESS COMMA
NO
.
X'4B'
YES
PROCESS DECIMAL
NO
(
X'4D'
YES
PROCESS L. PAREN
NO
)
X'5D'
YES
PROCESS R. PAREN
NO
NO
?
X'6F'
YES
PROCESS END REQ
EXIT
ENTER
NEXT CHARACTER
ADD 1 TO INPUT POINTER
END OF CARD
NO
EXIT
YES
CARD END OF FILE
YES
EXIT
NO
READ INPUT CARD
PRINT INPUT CARD
INCREMENT CARD COUNT INITIALIZE CARD POINTER
25 CARDS THIS REQUEST
NO
EXIT
YES
NOTE ERROR REQUEST TOO LONG
CARD EOF
DUMMY END OF REQUEST CHARACTER
EXIT

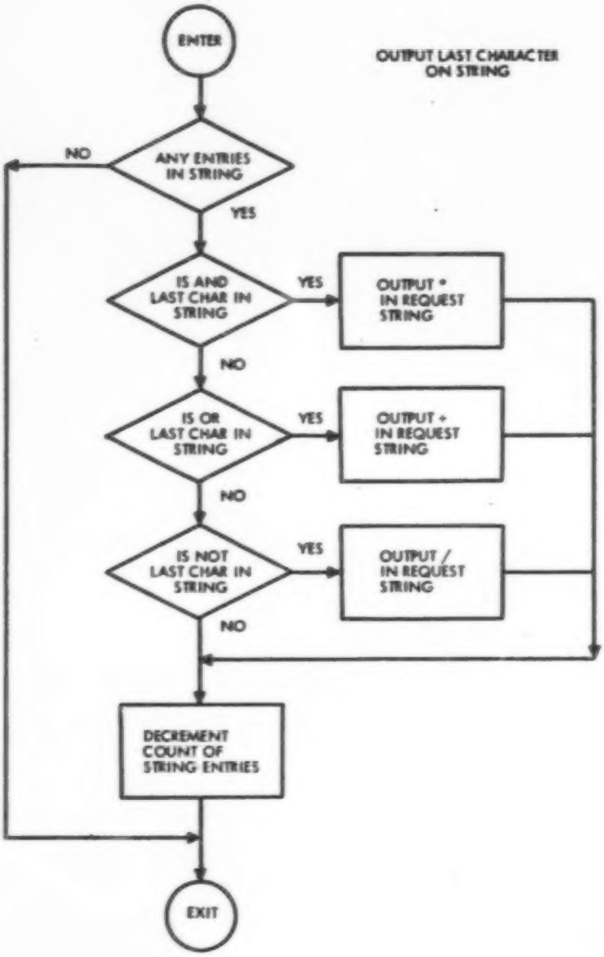

ENTER
PROCESS AND
ENTER
PROCESS NOT
INITIALIZE
INITIALIZE
SCAN OR LINK
NO
YES
NOTE ERROR
AND IS
INVALID
HERE
DATA LENGTH=O
YES
WAS R. PAREN LAST
NO
NO
YES
PROCESS
LAST
SEPARATOR
NOTE ERROR
INVALID
OPERATOR
OUTPUT LAST
CHARACTER
IN STRING
ANY
ENTRIES IN STRING
NO
YES
YES
IS "AND"
LAST IN STRING
NO
YES
IS "NOT"
LAST IN STRING
NO
PUT THIS SEPARATOR
ON END OF STRING
AND INCREMENT
EXIT
ENTER
PROCESS OR
INITIALIZE
YES
DATA LENGTH=O
WAS R. PAREN LAST
NO
NO
YES
PROCESS
LAST
SEPARATOR
NOTE ERROR
INVALID
OPERATOR
OUTPUT LAST
CHARACTER
IN STRING
ANY
ENTRIES IN STRING
NO
YES
YES
IS "AND"
LAST IN STRING
NO
YES
IS "NOT"
LAST IN STRING
NO
IS "OR"
LAST IN STRING
NO
YES
OUTPUT LAST
CHARACTER
IN STRING
PUT THIS SEPARATOR
ON END OF STRING
AND INCREMENT
EXIT


SEARCH
ENTER
PROCESS COMMA
INITIALIZE
DATA LENGTH=O
YES
NO
NOTE ERROR INVALID SEPARATOR
PROCESS LAST SEPARATOR
EXIT
ENTER
PROCESS END REQ
INITIALIZE
DATA LENGTH=O
YES
WAS COMMA OR R. PAREN LAST
YES
NO
NO
WAS R. PAREN OR L. PAREN LAST
YES
NO
PROCESS LAST SEPARATOR
NOTE ERROR INVALID SEPARATOR
ANY ENTRIES IN STRING
NO
YES
OUTPUT LAST CHARACTER IN STRING
EXIT
ENTER
PROCESS DECIMAL
INITIALIZE
DATA LENGTH=O
NOTE ERROR INVALID SEPARATOR
FLAG AS SINGLE VALUE
YES
FLAG AS DOUBLE VALUE
WAS DECIMAL LAST
NO
WAS COMMA LAST
NO
NOTE ERROR INVALID SEPARATOR
YES
EQ, NE, GE LE, GT, LT
NO
NOTE ERROR INVALID NUMERIC OPERATOR
YES
EQ=X'1E' NE=X'21'
GT=X'07' GE=X'1F'
LT =X'2C' LE=X'3E'
CONVERT VALUE TO FLOATING POINT
VALID NUMBER
NO
NOTE ERROR INVALID VALUE
YES
OUTPUT VALUE
EXIT
ENTER
PROCESS L. PAREN
INITIALIZE
SCAN OR LINK
YES
NOTE ERROR INVALID SEPARATOR
NO
DATA LENGTH=O
YES
NO
WAS COMMA, DECIMAL OR R. PAREN LAST
NO
YES
AND, NOT
WAS AND, OR OR NOT LAST
NO
OR
NOTE ERROR INVALID SEPARATOR
FC, FN, EC, EN,
YES
SET SCAN SWITCH
NO
LINK
YES
SET LINK SWITCH
NO
PROCESS LAST SEPARATOR
PUT THIS SEPARATOR ON END OF STRING AND INCREMENT
EXIT

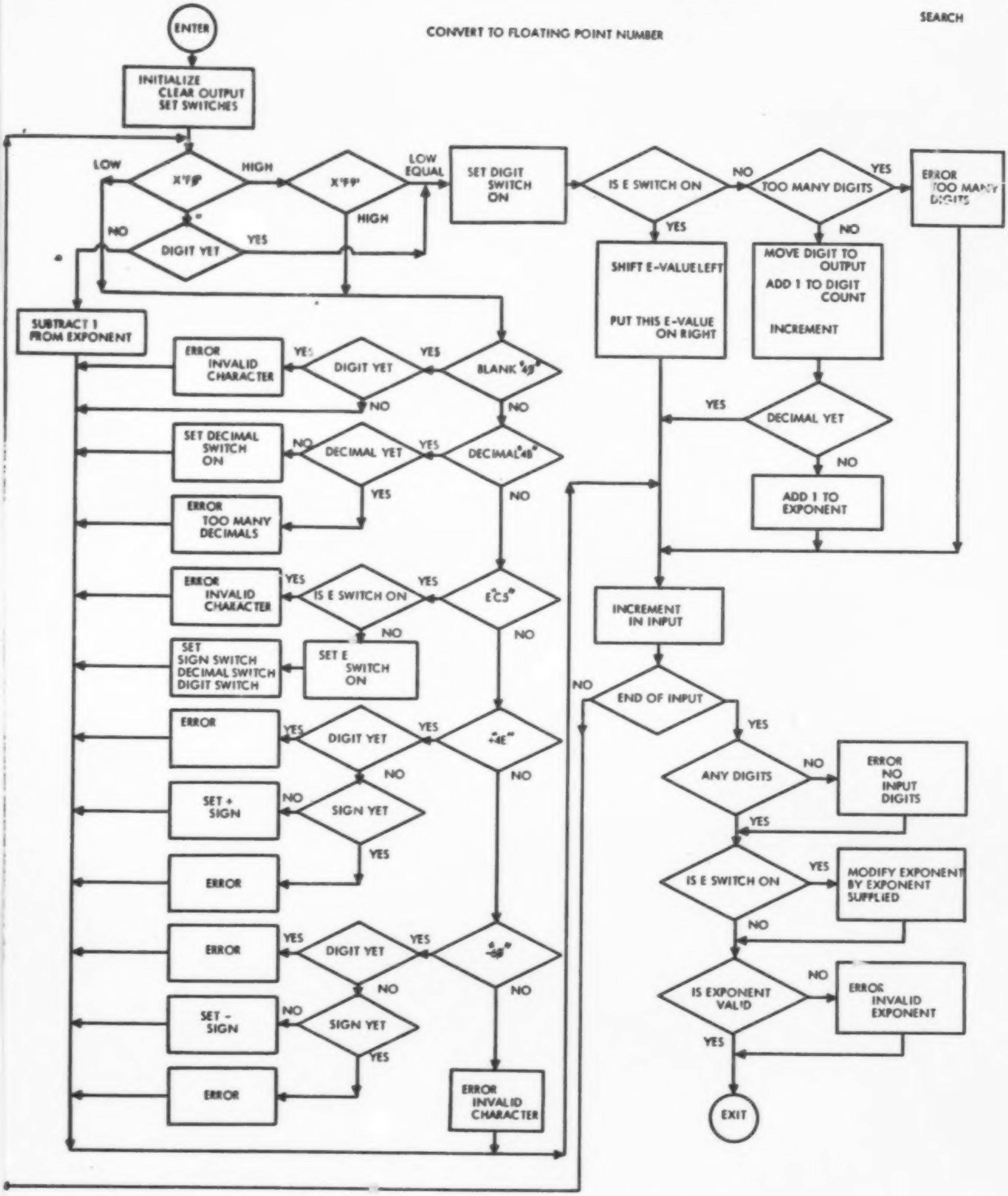

ENTER
PROCESS R. PAREN
INITIALIZE
DATA LENGTH=O
YES
WAS COMMA ON R. PAREN LAST
YES
NO
NO
WAS R. PAREN OR L. PAREN LAST
YES
NO
PROCESS LAST SEPARATOR
NOTE ERROR INVALID SEPARATOR
OUTPUT LAST CHARACTER IN STRING
ANY ENTRIES IN STRING
NO
YES
NO
IS L. PAREN LAST CHAR IN STRING
YES
DECREMENT COUNT OF STRING ENTRIES
IS SCAN SWITCH ON
YES
TURN SCAN SWITCH OFF
NO
IS LINK SWITCH ON
YES
TURN LINK SWITCH OFF
NO
EXIT

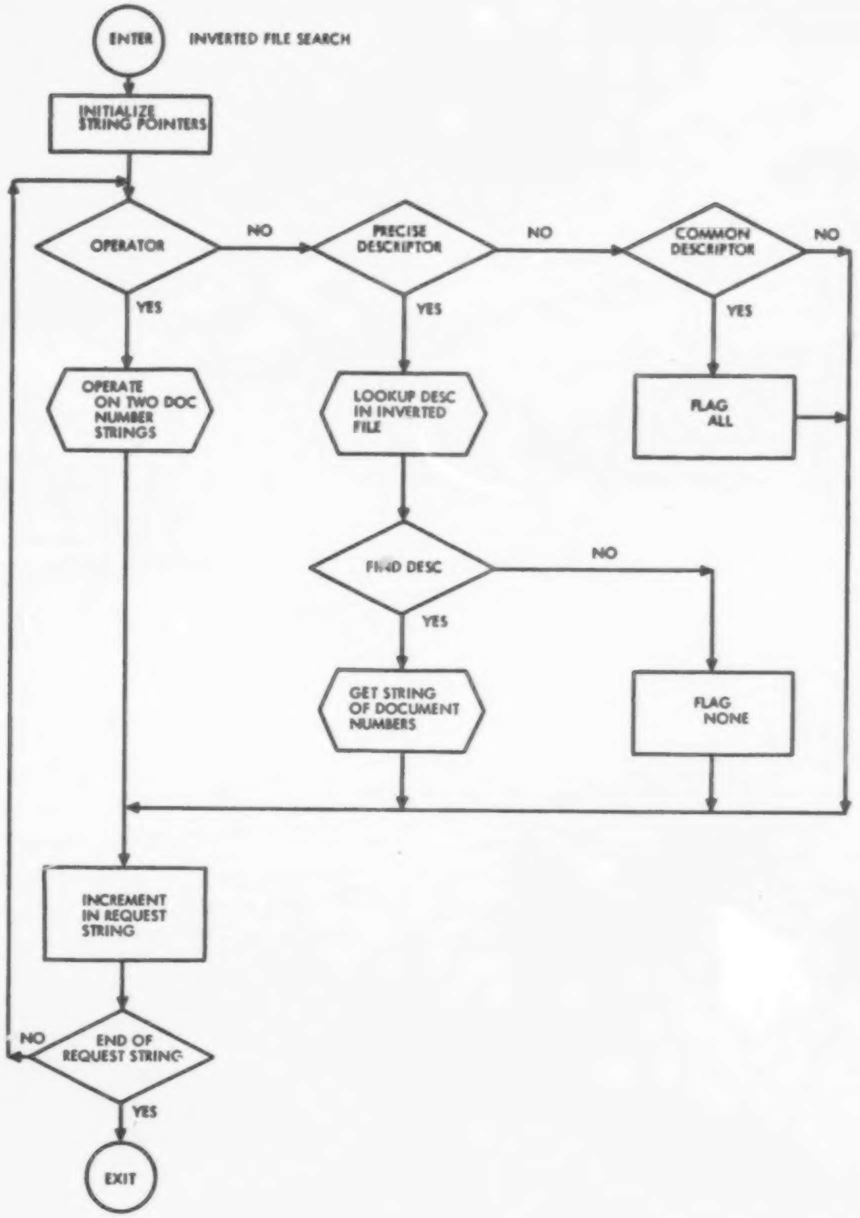

ENTER
PROCESS LAST SEPARATOR
WAS AND LAST
YES
NO
LOOKUP DESCRIPTOR
IS THIS SEPARATOR "OR"
NO
YES
WAS OR LAST
YES
NO
L. PAREN COUNT= R. PAREN COUNT
NO
YES
WAS NOT LAST
YES
NO
IS THIS DESCR A PRECISE DESC
YES
NO
WAS L. PAREN LAST
YES
NO
NOTE ERROR MUST BE ID FILE SEARCH
WAS COMMA LAST
YES
NO
LOOKUP SUB-DESCR
WAS R. PAREN LAST
YES
NO
NOTE ERROR INVALID SEPARATOR
WAS DECIMAL LAST
YES
NO
WAS A VALUE LAST
YES
NO
FIRST TIME INITIALIZE
NOTE ERROR INVALID VALUE
EXIT
ENTER
LOOKUP DESCRIPTOR
LOOKUP SUB-DESCR
INITIALIZE
WAS LAST CHARACTER A BLANK
NO
YES
DECREMENT LENGTH BY ONE
TRUNCATION
NO
YES
SET TRUNCATION SWITCH
LOOK UP DESC OR SD ON DICT DISK
INSERT THIS USE TERM AS DESC
FIND IT
NO
YES
NOTE ERROR CANNOT FIND DESC
YES
USE TERM
NO
OUTPUT DESC IN REQUEST STRING
EXIT

OUTPUT LAST CHARACTER
ON STRING

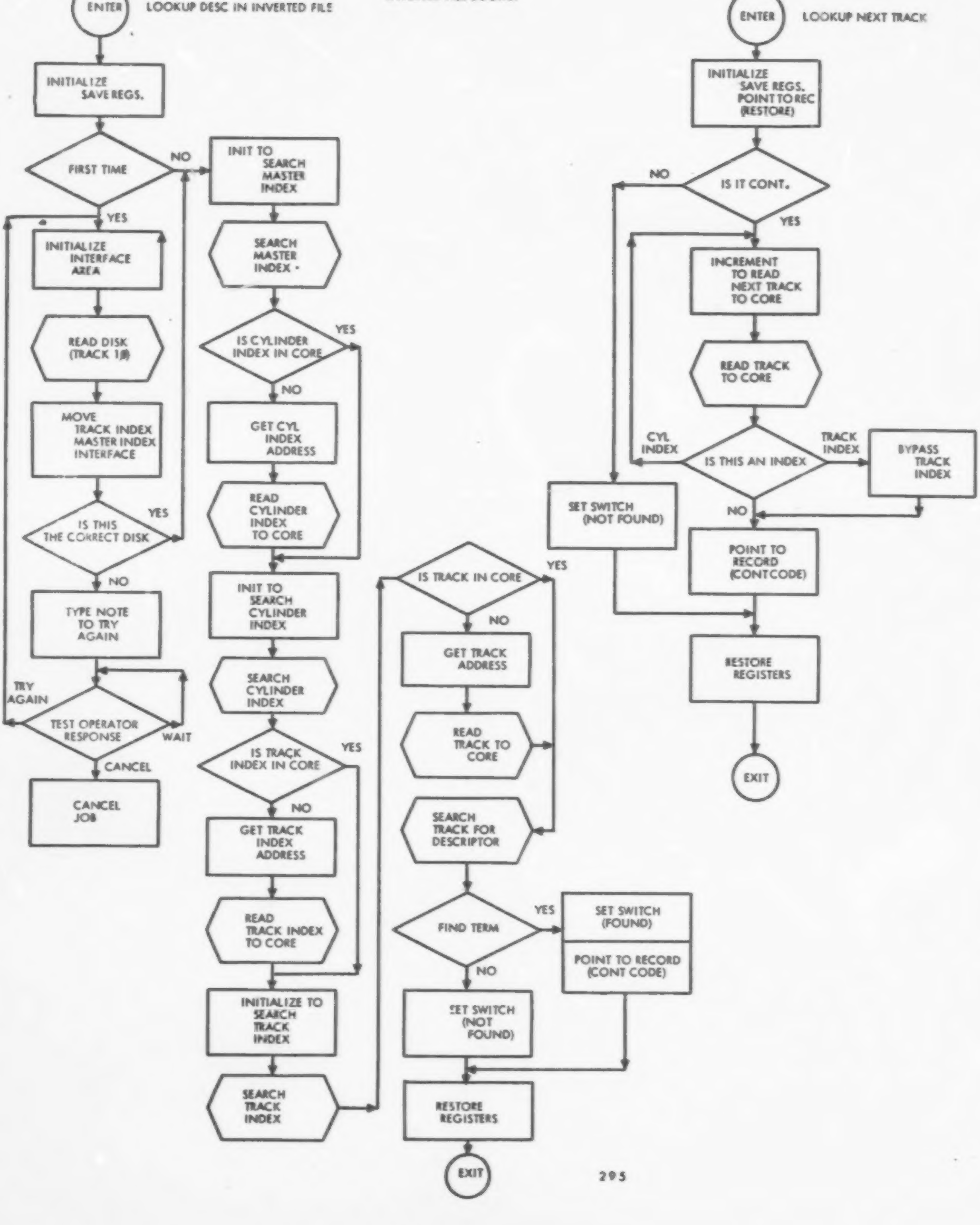

ENTER
ANY ENTRIES IN STRING
NO
YES
IS AND LAST CHAR IN STRING
YES
OUTPUT * IN REQUEST STRING
NO
IS OR LAST CHAR IN STRING
YES
OUTPUT + IN REQUEST STRING
NO
IS NOT LAST CHAR IN STRING
YES
OUTPUT / IN REQUEST STRING
NO
DECREMENT COUNT OF STRING ENTRIES
EXIT

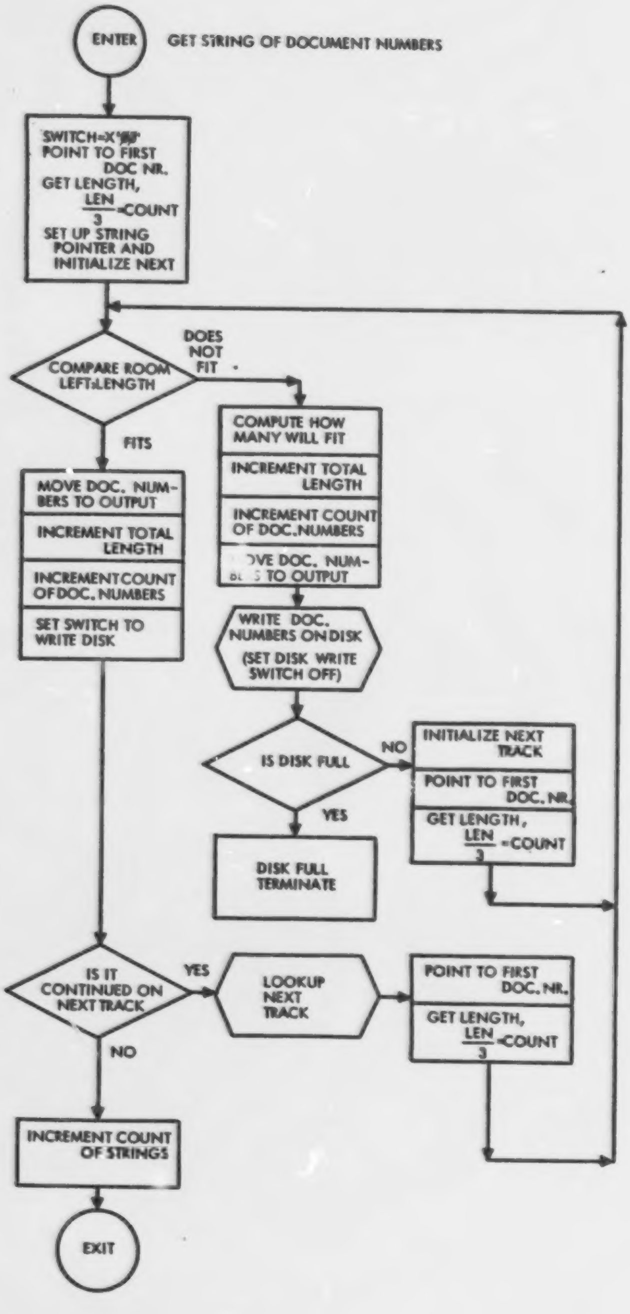
CONVERT TO FLOATING POINT NUMBER

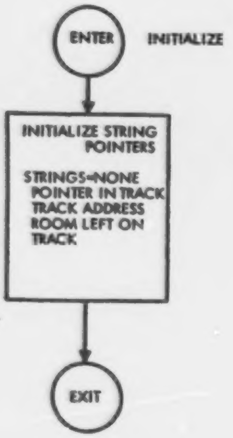

ENTER
INITIALIZE
CLEAR OUTPUT
SET SWITCHES
X'F0'
LOW
HIGH
X'F9'
LOW
EQUAL
HIGH
NO
DIGIT YET
YES
SET DIGIT SWITCH ON
IS E SWITCH ON
NO
YES
TOO MANY DIGITS
YES
NO
ERROR TOO MANY DIGITS
SHIFT E-VALUE LEFT
PUT THIS E-VALUE ON RIGHT
MOVE DIGIT TO OUTPUT
ADD 1 TO DIGIT COUNT
INCREMENT
SUBTRACT 1 FROM EXPONENT
ERROR INVALID CHARACTER
YES
DIGIT YET
YES
BLANK '40'
NO
NO
YES
DECIMAL YET
NO
SET DECIMAL SWITCH ON
NO
DECIMAL YET
YES
DECIMAL '4B'
YES
NO
ADD 1 TO EXPONENT
ERROR TOO MANY DECIMALS
ERROR INVALID CHARACTER
YES
IS E SWITCH ON
YES
E 'C5'
NO
NO
INCREMENT IN INPUT
SET SIGN SWITCH DECIMAL SWITCH DIGIT SWITCH
SET E SWITCH ON
NO
END OF INPUT
YES
ERROR
YES
DIGIT YET
YES
'4E'
NO
NO
ANY DIGITS
NO
ERROR NO INPUT DIGITS
SET + SIGN
NO
SIGN YET
YES
YES
IS E SWITCH ON
YES
MODIFY EXPONENT BY EXPONENT SUPPLIED
ERROR
NO
ERROR
YES
DIGIT YET
YES
'60'
NO
NO
IS EXPONENT VALID
NO
ERROR INVALID EXPONENT
SET - SIGN
NO
SIGN YET
YES
YES
ERROR
ERROR INVALID CHARACTER
EXIT

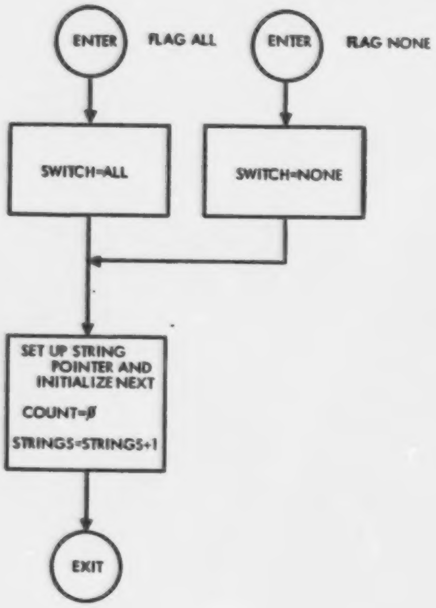

ENTER
INVERTED FILE SEARCH
INITIALIZE STRING POINTERS
OPERATOR
NO
YES
OPERATE ON TWO DOC NUMBER STRINGS
PRECISE DESCRIPTOR
NO
YES
LOOKUP DESC IN INVERTED FILE
FIND DESC
NO
YES
GET STRING OF DOCUMENT NUMBERS
FLAG NONE
COMMON DESCRIPTOR
NO
YES
FLAG ALL
INCREMENT IN REQUEST STRING
END OF REQUEST STRING
NO
YES
EXIT

INVERTED FILE LOOKUP

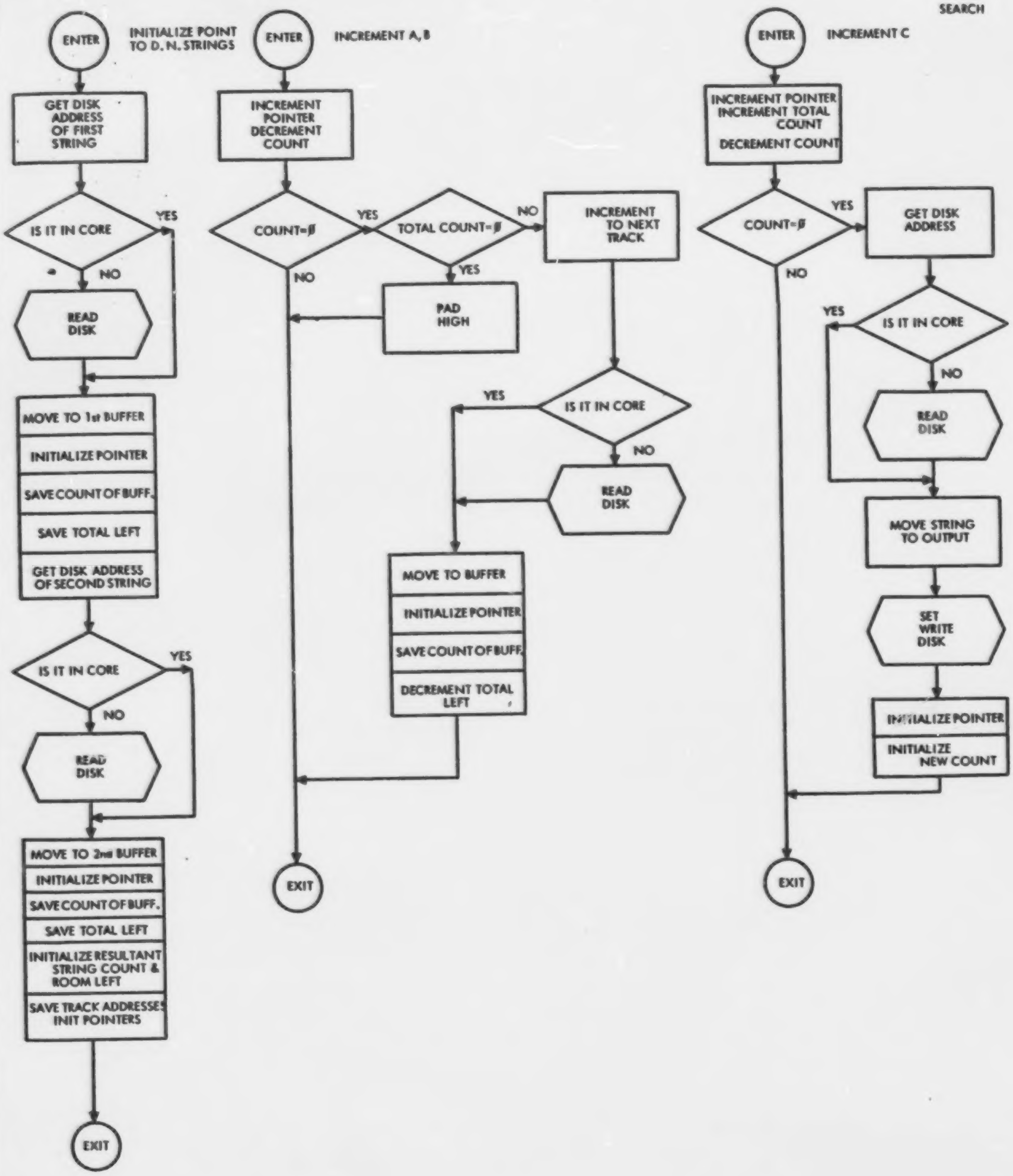

ENTER
LOOKUP DESC IN INVERTED FILE
INITIALIZE SAVE REGS.
FIRST TIME
NO
YES
INITIALIZE INTERFACE AREA
READ DISK (TRACK 10)
MOVE TRACK INDEX MASTER INDEX INTERFACE
IS THIS THE CORRECT DISK
YES
NO
TYPE NOTE TO TRY AGAIN
TRY AGAIN
TEST OPERATOR RESPONSE
WAIT
CANCEL
CANCEL JOB
INIT TO SEARCH MASTER INDEX
SEARCH MASTER INDEX
IS CYLINDER INDEX IN CORE
YES
NO
GET CYL INDEX ADDRESS
READ CYLINDER INDEX TO CORE
INIT TO SEARCH CYLINDER INDEX
SEARCH CYLINDER INDEX
IS TRACK INDEX IN CORE
YES
NO
GET TRACK INDEX ADDRESS
READ TRACK INDEX TO CORE
INITIALIZE TO SEARCH TRACK INDEX
SEARCH TRACK INDEX
IS TRACK IN CORE
YES
NO
GET TRACK ADDRESS
READ TRACK TO CORE
SEARCH TRACK FOR DESCRIPTOR
FIND TERM
YES
SET SWITCH (FOUND)
POINT TO RECORD (CONT CODE)
NO
SET SWITCH (NOT FOUND)
RESTORE REGISTERS
EXIT
ENTER
LOOKUP NEXT TRACK
INITIALIZE SAVE REGS. POINT TO REC (RESTORE)
IS IT CONT.
NO
YES
INCREMENT TO READ NEXT TRACK TO CORE
READ TRACK TO CORE
IS THIS AN INDEX
CYL INDEX
TRACK INDEX
BYPASS TRACK INDEX
NO
SET SWITCH (NOT FOUND)
POINT TO RECORD (CONT CODE)
RESTORE REGISTERS
EXIT

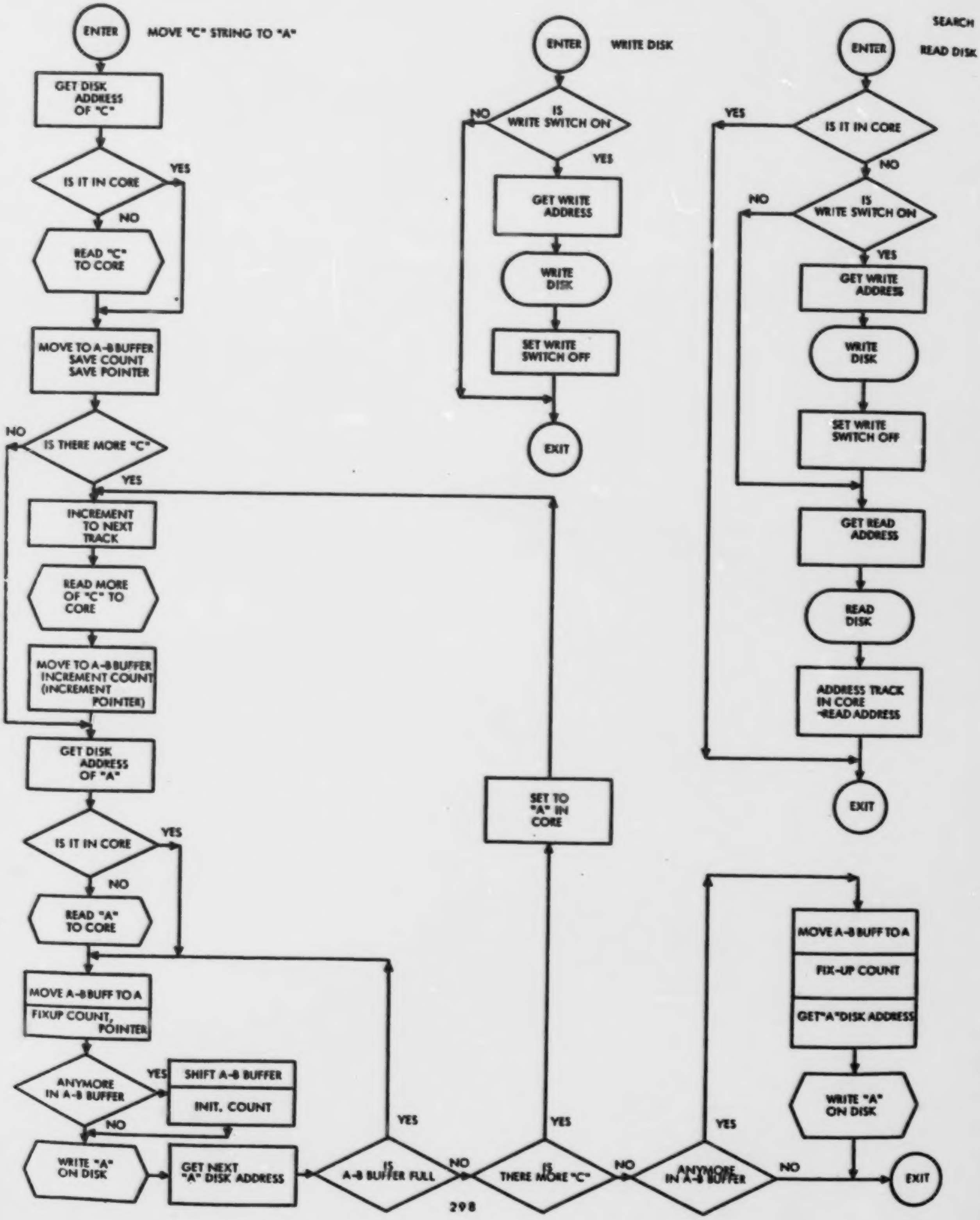

ENTER
GET STRING OF DOCUMENT NUMBERS
SWITCH=X'00' POINT TO FIRST DOC NR. GET LENGTH, LEN/3=COUNT SET UP STRING POINTER AND INITIALIZE NEXT
COMPARE ROOM LEFT:LENGTH
DOES NOT FIT
FITS
MOVE DOC. NUMBERS TO OUTPUT
INCREMENT TOTAL LENGTH
INCREMENT COUNT OF DOC. NUMBERS
SET SWITCH TO WRITE DISK
COMPUTE HOW MANY WILL FIT
INCREMENT TOTAL LENGTH
INCREMENT COUNT OF DOC.NUMBERS
MOVE DOC. NUMBERS TO OUTPUT
WRITE DOC. NUMBERS ON DISK (SET DISK WRITE SWITCH OFF)
IS DISK FULL
NO
YES
DISK FULL TERMINATE
INITIALIZE NEXT TRACK
POINT TO FIRST DOC. NR.
GET LENGTH, LEN/3 =COUNT
IS IT CONTINUED ON NEXT TRACK
YES
NO
LOOKUP NEXT TRACK
POINT TO FIRST DOC. NR.
GET LENGTH, LEN/3 =COUNT
INCREMENT COUNT OF STRINGS
EXIT


ENTER
INITIALIZE
INITIALIZE STRING POINTERS
STRINGS=NONE
POINTER IN TRACK
TRACK ADDRESS
ROOM LEFT ON TRACK
EXIT

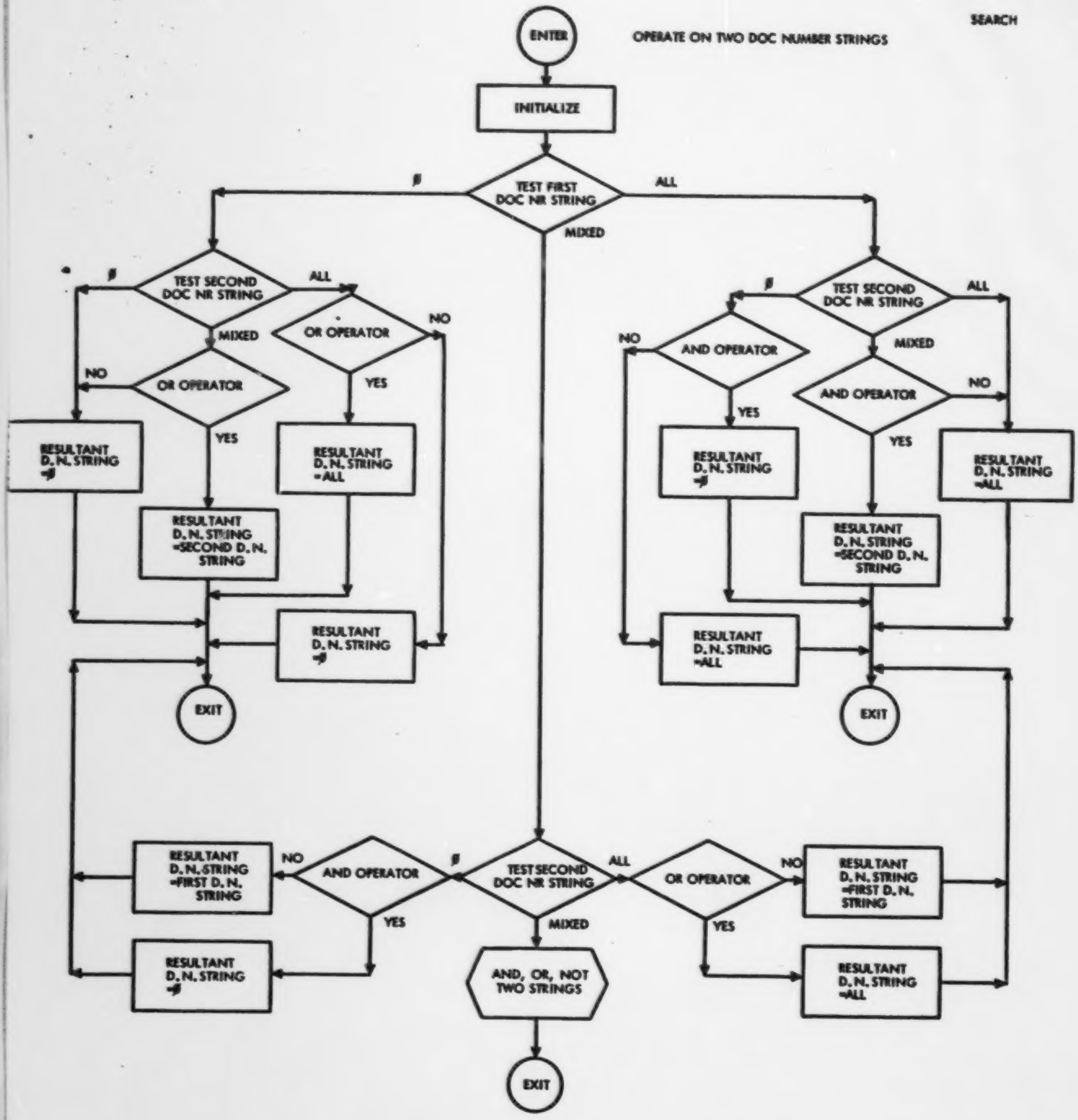

ENTER
FLAG ALL
ENTER
FLAG NONE
SWITCH=ALL
SWITCH=NONE
SET UP STRING POINTER AND INITIALIZE NEXT
COUNT=0
STRINGS=STRINGS+1
EXIT

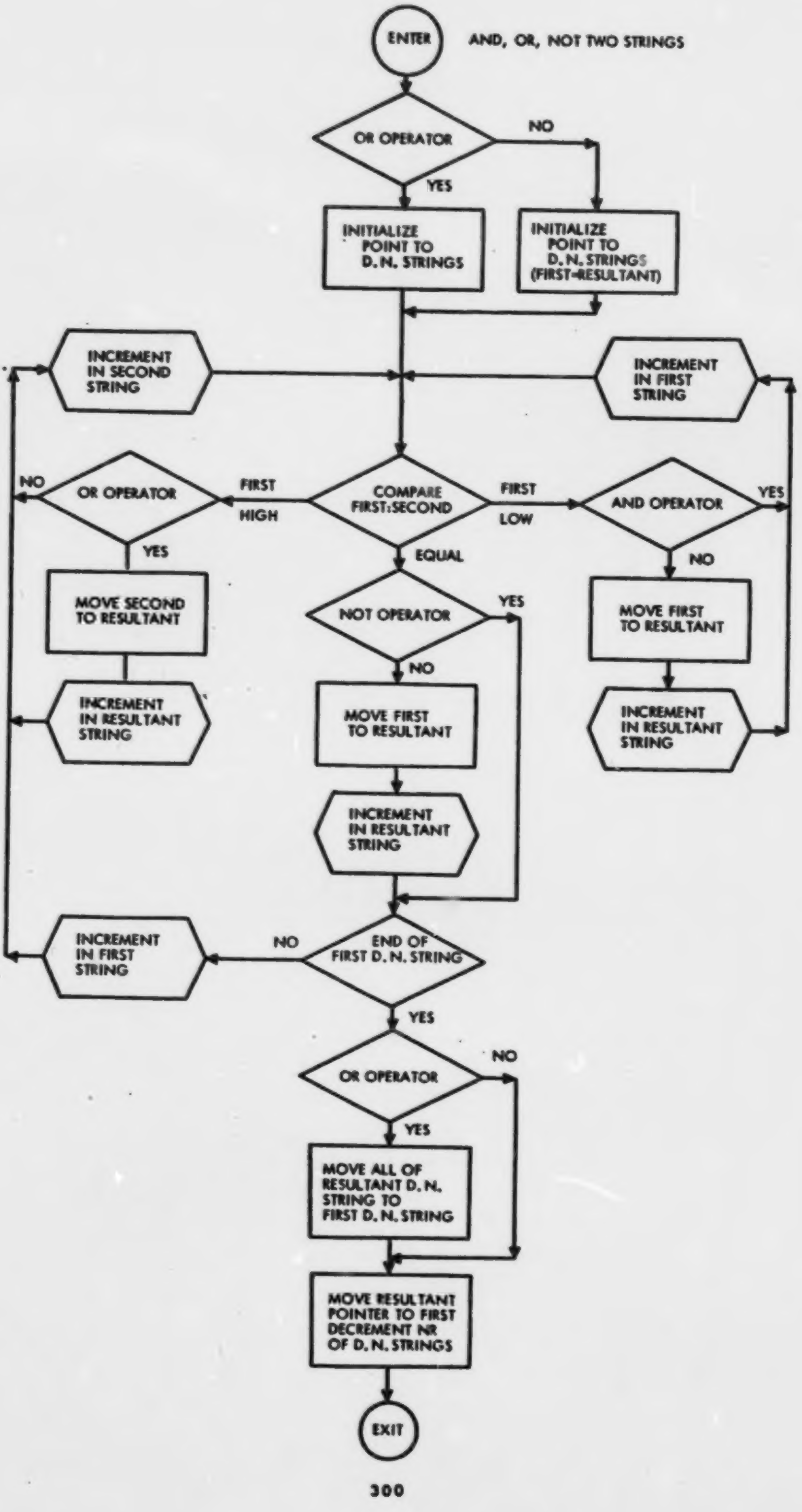
SEARCH


ENTER
INITIALIZE POINT TO D. N. STRINGS
GET DISK ADDRESS OF FIRST STRING
IS IT IN CORE
YES
NO
READ DISK
MOVE TO 1st BUFFER
INITIALIZE POINTER
SAVE COUNT OF BUFF.
SAVE TOTAL LEFT
GET DISK ADDRESS OF SECOND STRING
IS IT IN CORE
YES
NO
READ DISK
MOVE TO 2nd BUFFER
INITIALIZE POINTER
SAVE COUNT OF BUFF.
SAVE TOTAL LEFT
INITIALIZE RESULTANT STRING COUNT & ROOM LEFT
SAVE TRACK ADDRESSES INIT POINTERS
EXIT
ENTER
INCREMENT A, B
INCREMENT POINTER DECREMENT COUNT
COUNT=0
YES
NO
TOTAL COUNT=0
NO
YES
PAD HIGH
INCREMENT TO NEXT TRACK
IS IT IN CORE
YES
NO
READ DISK
MOVE TO BUFFER
INITIALIZE POINTER
SAVE COUNT OF BUFF.
DECREMENT TOTAL LEFT
EXIT
ENTER
INCREMENT C
INCREMENT POINTER INCREMENT TOTAL COUNT
DECREMENT COUNT
COUNT=0
YES
NO
GET DISK ADDRESS
IS IT IN CORE
YES
NO
READ DISK
MOVE STRING TO OUTPUT
SET WRITE DISK
INITIALIZE POINTER
INITIALIZE NEW COUNT
EXIT

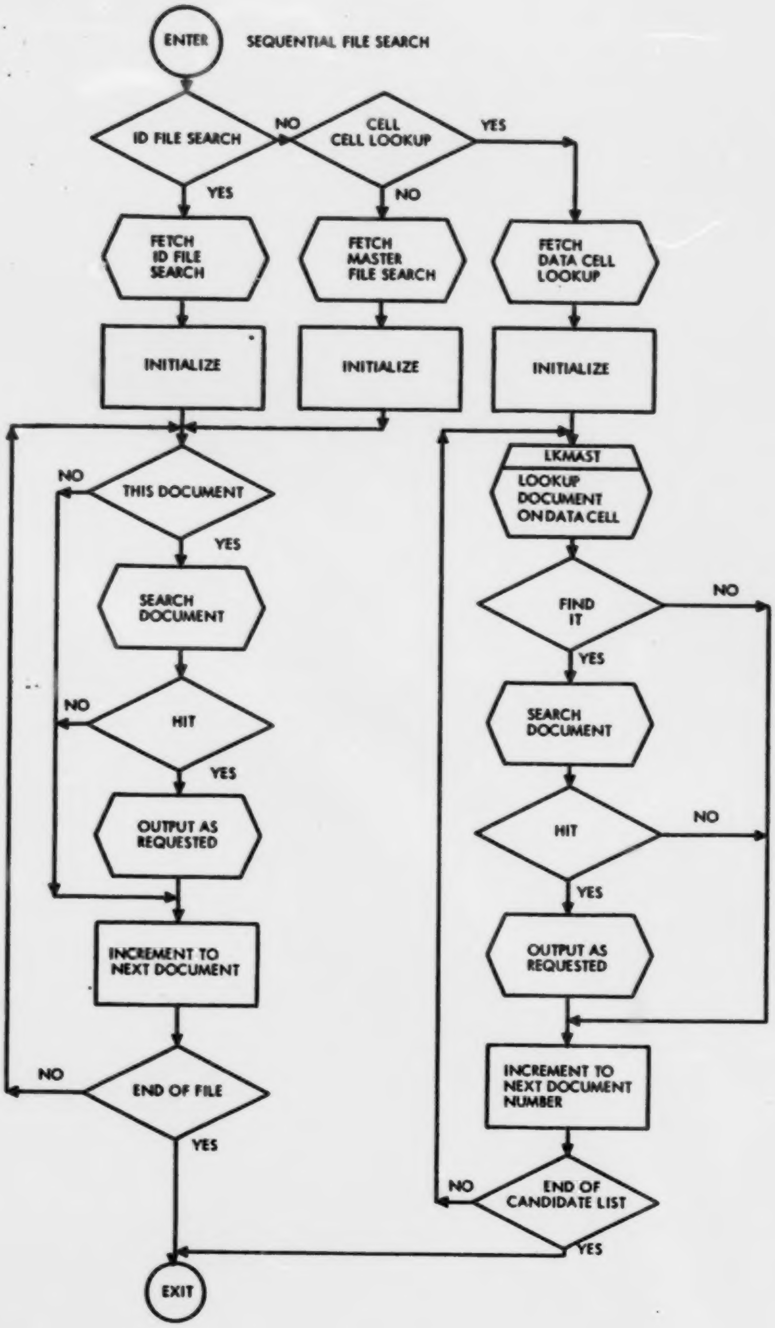

ENTER
MOVE "C" STRING TO "A"
GET DISK ADDRESS OF "C"
IS IT IN CORE
YES
NO
READ "C" TO CORE
MOVE TO A-B BUFFER SAVE COUNT SAVE POINTER
NO
IS THERE MORE "C"
YES
INCREMENT TO NEXT TRACK
READ MORE OF "C" TO CORE
MOVE TO A-B BUFFER INCREMENT COUNT (INCREMENT POINTER)
GET DISK ADDRESS OF "A"
IS IT IN CORE
YES
NO
READ "A" TO CORE
MOVE A-B BUFF TO A
FIXUP COUNT, POINTER
ANYMORE IN A-B BUFFER
YES
SHIFT A-B BUFFER
INIT. COUNT
NO
WRITE "A" ON DISK
GET NEXT "A" DISK ADDRESS
IS A-B BUFFER FULL
YES
NO
IS THERE MORE "C"
YES
NO
SET TO "A" IN CORE
ANYMORE IN A-B BUFFER
YES
NO
MOVE A-B BUFF TO A
FIX-UP COUNT
GET "A" DISK ADDRESS
WRITE "A" ON DISK
EXIT
ENTER
WRITE DISK
IS WRITE SWITCH ON
NO
YES
GET WRITE ADDRESS
WRITE DISK
SET WRITE SWITCH OFF
EXIT
SEARCH
ENTER
READ DISK
IS IT IN CORE
YES
NO
IS WRITE SWITCH ON
NO
YES
GET WRITE ADDRESS
WRITE DISK
SET WRITE SWITCH OFF
GET READ ADDRESS
READ DISK
ADDRESS TRACK IN CORE →READ ADDRESS
EXIT

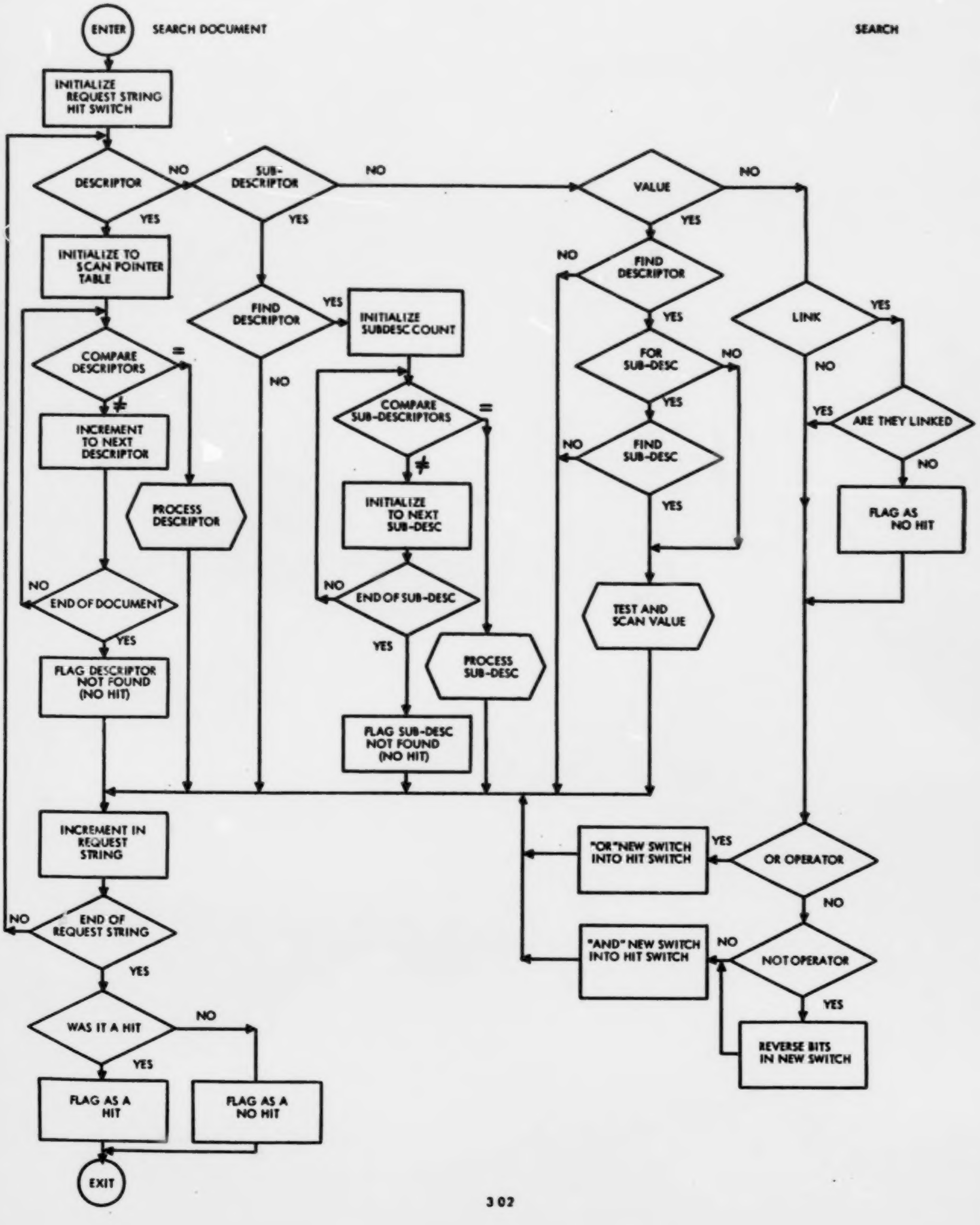
OPERATE ON TWO DOC NUMBER STRINGS

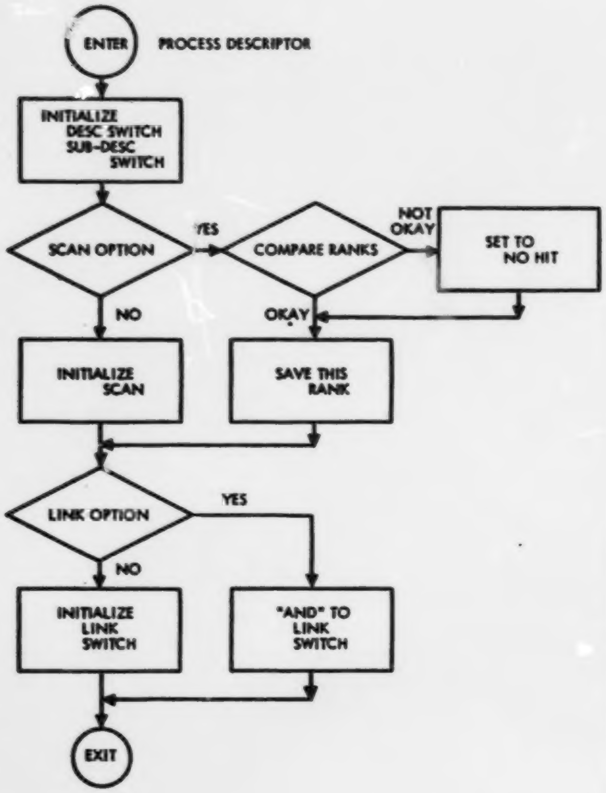

ENTER
INITIALIZE
TEST FIRST DOC NR STRING
0
ALL
MIXED
TEST SECOND DOC NR STRING
0
ALL
MIXED
OR OPERATOR
NO
YES
OR OPERATOR
NO
YES
RESULTANT D. N. STRING =0
RESULTANT D. N. STRING =SECOND D. N. STRING
RESULTANT D. N. STRING =ALL
RESULTANT D. N. STRING =0
EXIT
TEST SECOND DOC NR STRING
0
ALL
MIXED
AND OPERATOR
NO
YES
AND OPERATOR
NO
YES
RESULTANT D. N. STRING =0
RESULTANT D. N. STRING =SECOND D. N. STRING
RESULTANT D. N. STRING =ALL
RESULTANT D. N. STRING =ALL
EXIT
TEST SECOND DOC NR STRING
0
ALL
MIXED
AND OPERATOR
NO
YES
RESULTANT D. N. STRING =FIRST D. N. STRING
RESULTANT D. N. STRING =0
OR OPERATOR
NO
YES
RESULTANT D. N. STRING =FIRST D. N. STRING
RESULTANT D. N. STRING =ALL
AND, OR, NOT TWO STRINGS
EXIT

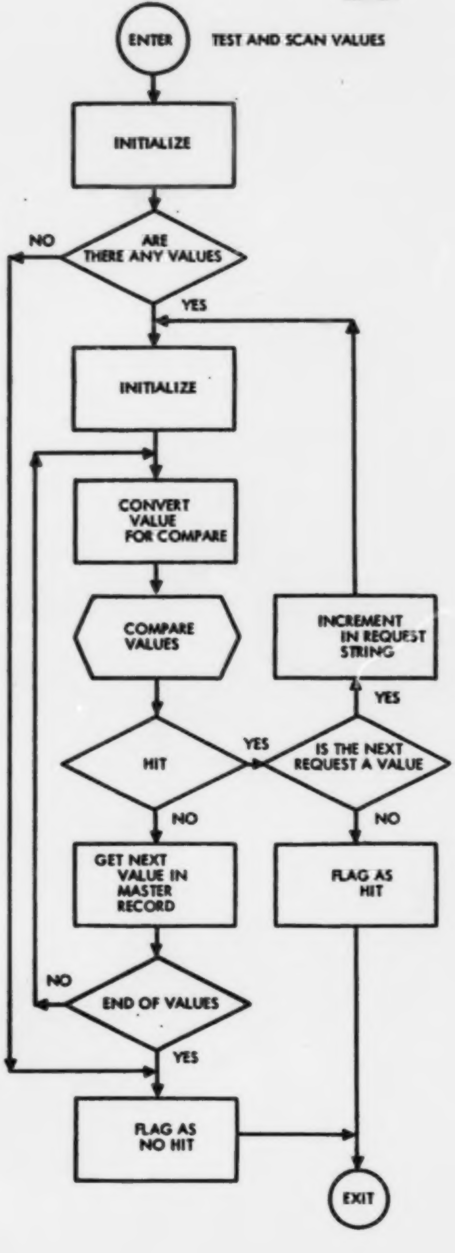

ENTER
AND, OR, NOT TWO STRINGS
OR OPERATOR
NO
YES
INITIALIZE POINT TO D. N. STRINGS
INITIALIZE POINT TO D. N. STRINGS (FIRST=RESULTANT)
INCREMENT IN SECOND STRING
INCREMENT IN FIRST STRING
NO
OR OPERATOR
FIRST HIGH
COMPARE FIRST:SECOND
FIRST LOW
AND OPERATOR
YES
YES
EQUAL
NO
MOVE SECOND TO RESULTANT
NOT OPERATOR
YES
MOVE FIRST TO RESULTANT
NO
INCREMENT IN RESULTANT STRING
MOVE FIRST TO RESULTANT
INCREMENT IN RESULTANT STRING
INCREMENT IN RESULTANT STRING
INCREMENT IN FIRST STRING
NO
END OF FIRST D. N. STRING
YES
OR OPERATOR
NO
YES
MOVE ALL OF RESULTANT D. N. STRING TO FIRST D. N. STRING
MOVE RESULTANT POINTER TO FIRST DECREMENT NR OF D. N. STRINGS
EXIT

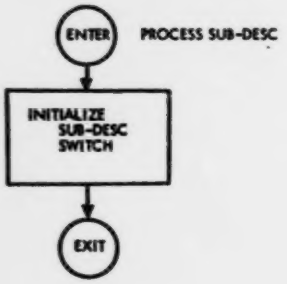
SEQUENTIAL FILE SEARCH

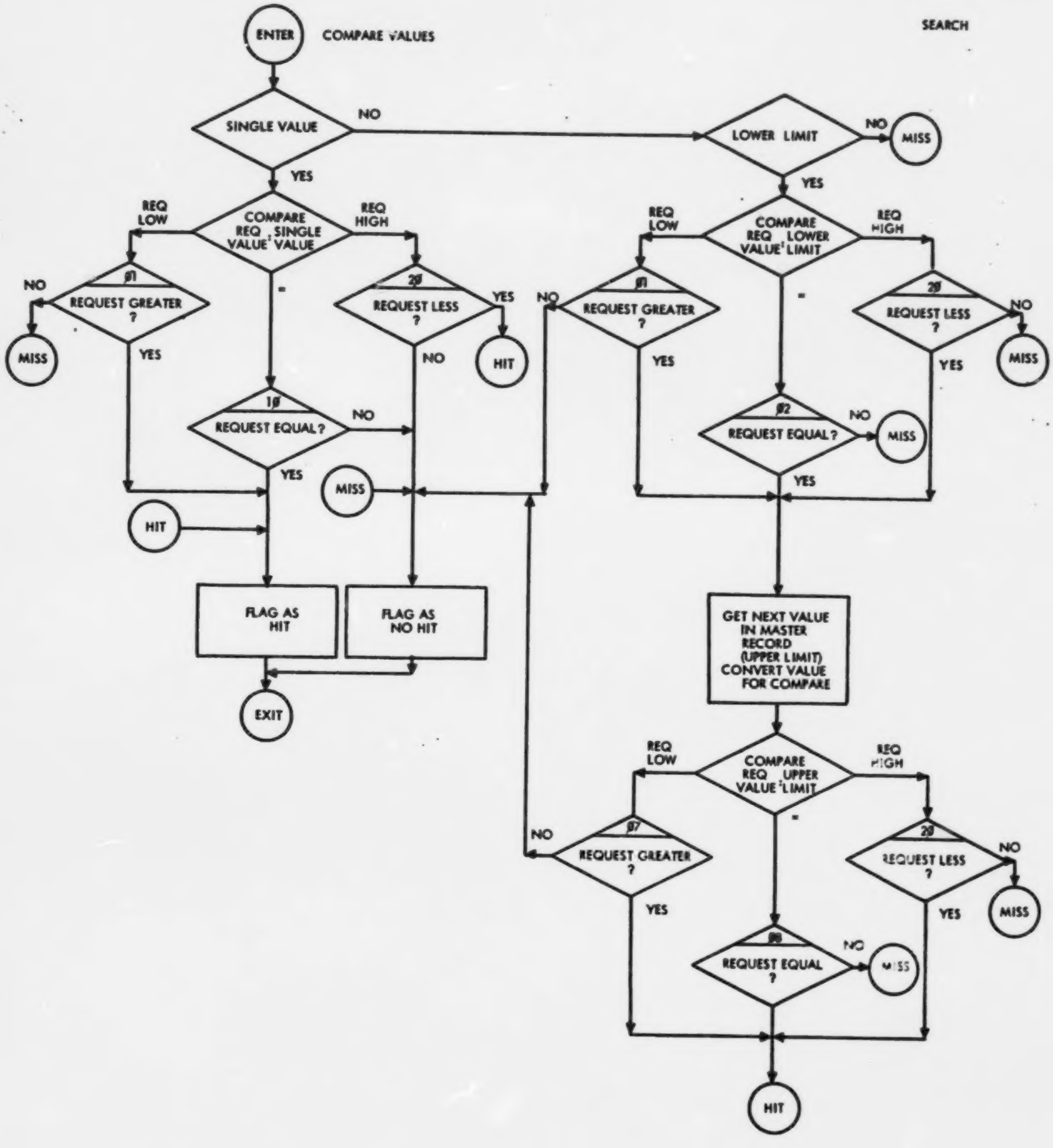

ENTER
ID FILE SEARCH
NO
CELL
CELL LOOKUP
YES
YES
NO
FETCH
ID FILE
SEARCH
FETCH
MASTER
FILE SEARCH
FETCH
DATA CELL
LOOKUP
INITIALIZE
INITIALIZE
INITIALIZE
NO
THIS DOCUMENT
YES
LKMAST
LOOKUP
DOCUMENT
ON DATA CELL
SEARCH
DOCUMENT
FIND
IT
NO
YES
NO
HIT
YES
SEARCH
DOCUMENT
OUTPUT AS
REQUESTED
HIT
NO
YES
OUTPUT AS
REQUESTED
INCREMENT TO
NEXT DOCUMENT
INCREMENT TO
NEXT DOCUMENT
NUMBER
NO
END OF FILE
YES
NO
END OF
CANDIDATE LIST
YES
EXIT


ENTER
SEARCH DOCUMENT
SEARCH
INITIALIZE REQUEST STRING HIT SWITCH
DESCRIPTOR
NO
YES
SUB-DESCRIPTOR
NO
YES
VALUE
NO
YES
INITIALIZE TO SCAN POINTER TABLE
FIND DESCRIPTOR
YES
NO
INITIALIZE SUBDESC COUNT
FIND DESCRIPTOR
NO
YES
LINK
YES
NO
COMPARE DESCRIPTORS
=
≠
INCREMENT TO NEXT DESCRIPTOR
PROCESS DESCRIPTOR
COMPARE SUB-DESCRIPTORS
=
≠
INITIALIZE TO NEXT SUB-DESC
FOR SUB-DESC
NO
YES
FIND SUB-DESC
NO
YES
ARE THEY LINKED
YES
NO
FLAG AS NO HIT
END OF DOCUMENT
NO
YES
END OF SUB-DESC
NO
YES
TEST AND SCAN VALUE
PROCESS SUB-DESC
FLAG DESCRIPTOR NOT FOUND (NO HIT)
FLAG SUB-DESC NOT FOUND (NO HIT)
INCREMENT IN REQUEST STRING
"OR" NEW SWITCH INTO HIT SWITCH
YES
OR OPERATOR
NO
END OF REQUEST STRING
NO
YES
"AND" NEW SWITCH INTO HIT SWITCH
NO
NOT OPERATOR
YES
REVERSE BITS IN NEW SWITCH
WAS IT A HIT
NO
YES
FLAG AS A HIT
FLAG AS A NO HIT
EXIT


ENTER
PROCESS DESCRIPTOR
INITIALIZE DESC SWITCH SUB-DESC SWITCH
SCAN OPTION
YES
COMPARE RANKS
NOT OKAY
SET TO NO HIT
OKAY
NO
INITIALIZE SCAN
SAVE THIS RANK
LINK OPTION
YES
NO
INITIALIZE LINK SWITCH
"AND" TO LINK SWITCH
EXIT


ENTER
TEST AND SCAN VALUES
INITIALIZE
NO
ARE THERE ANY VALUES
YES
INITIALIZE
CONVERT VALUE FOR COMPARE
COMPARE VALUES
INCREMENT IN REQUEST STRING
YES
HIT
YES
IS THE NEXT REQUEST A VALUE
NO
NO
GET NEXT VALUE IN MASTER RECORD
FLAG AS HIT
NO
END OF VALUES
YES
FLAG AS NO HIT
EXIT


ENTER
PROCESS SUB-DESC
INITIALIZE SUB-DESC SWITCH
EXIT


ENTER
COMPARE VALUES
SINGLE VALUE
NO
YES
LOWER LIMIT
NO
MISS
YES
REQ LOW
COMPARE REQ VALUE : SINGLE VALUE
REQ HIGH
=
01
REQUEST GREATER ?
NO
MISS
YES
20
REQUEST LESS ?
YES
HIT
NO
10
REQUEST EQUAL?
NO
YES
MISS
HIT
FLAG AS HIT
FLAG AS NO HIT
EXIT
REQ LOW
COMPARE REQ VALUE : LOWER LIMIT
REQ HIGH
=
NO
01
REQUEST GREATER ?
YES
20
REQUEST LESS ?
NO
MISS
YES
02
REQUEST EQUAL ?
NO
MISS
YES
GET NEXT VALUE IN MASTER RECORD (UPPER LIMIT) CONVERT VALUE FOR COMPARE
REQ LOW
COMPARE REQ VALUE : UPPER LIMIT
REQ HIGH
=
NO
07
REQUEST GREATER ?
YES
20
REQUEST LESS ?
NO
MISS
YES
08
REQUEST EQUAL ?
NO
MISS
HIT

APPENDICES

APPENDIX A - DICTIONARY RECORD

APPENDIX B - MASTER FILE RECORD

- a. Searchable Section
- b. Free Text Segment

APPENDIX C - INVERTED FILE RECORD

APPENDIX D - CROSS REFERENCE RECORD

APPENDIX E - SEARCH TAPE OUTPUT RECORD FORMAT

APPENDIX A

DICTIONARY RECORD

LENGTH	CONTENTS
2 bytes	Record Length
2 bytes	Binary Zeros
20 bytes	Unused
20 bytes	Dictionary Codes
* 2 bytes	Disk Record Length
2 bytes	Binary Zeros
1 byte	Record Content Code
1 byte	Main Term Function
1 byte	Number of File/Context Pairs for Main Term
0-510 bytes	File/Context Codes
1 byte	Main Term Length +3
1 byte	Main Term Length +2
1 byte	Main Term Code
1≤N≤64 bytes	Main Term
1 byte	Use Term Length +3
1 byte	Use Term Length +2
1 byte	Use Term Code
1≤N≤64 bytes	Use Term -1 Only
1 byte	Related Term + File/Context +3 Length
1 byte	Related Term Length +2
1 byte	Related Term Code
1≤N≤64 bytes	Related Term
0-510 bytes	Related Term File/Context Codes
1 byte	See Also Term Length +3
1 byte	See Also Term Length +2
1 byte	See Also Term Code
1≤N≤64 bytes	See Also Term
1 byte	Scope Note Length +4
1 byte	Scope Note Length +3
1 byte	Scope Note Code
1 byte	Scope Note Line Number
1≤N≤60 bytes	Scope Note

* The disk file version of the dictionary record does not include the first 44 bytes. It begins with the field labelled DISK RECORD LENGTH and concludes with either the USE term or the last RELATED term.

APPENDIX B

MASTER FILE RECORD

A. Searchable Section

LENGTH	CONTENTS	
1 byte	Continuation Flag	
2 bytes	Record Length	
3 bytes	Internal Number	
1 byte	Segment Number	
1 byte	Tag	
2 bytes	Run Number	
9 bytes	External Number	
1 byte	Security Code	
1 byte	File Number	
24 bytes	Title Information	
1 byte	Number of Pointers	
2 bytes	Displacement	Pointer to Descriptor Information. 255 Allowed
1 byte	Length of Descriptor	
1 byte	Rank of Descirptor	
1 byte	Type of Descriptor	
1 byte	Number of Context Codes	
1 byte	Number of Qualifiers	
1 to 64 bytes	Descriptor	
1 byte each	Context Code - 255 Allowed	
1 byte	Subdescriptor Length +2	
1 byte	Number of Values/Links	
0 to 64 bytes	Subdescriptor	
1 byte	Link/Value Code	
2 bytes	Link	
1 byte	Value Characteristic	
4 bytes	Value Mantissa	

B. Free Text Segment

LENGTH	CONTENTS
1 byte	Continuation Flag
2 bytes	Record Length
3 bytes	Internal Number
1 byte	Segment Number
1 byte	Tag
1 byte	Text Line Length +2
1 byte	Text Line Number
1 to 71 bytes	Text Line
1 byte	Text Line Length +2
1 byte	Text Line Number
1 to 71 bytes	Text Line

The ID. file record is in the same format as the master file record. The ID. file contains no free text.

APPENDIX C

INVERTED FILE RECORD

LENGTH	CONTENTS
2 bytes	Entire Record Length
2 bytes	Binary Zeros
2 bytes	String Number
1 byte	Continuation Flag
3 bytes	Highest Document Number this String
2 bytes	Total Document Numbers this String
6 bytes	Date Last Addition
6 bytes	Date Last Deletion
1 byte	Descriptor Function
* 1 byte	Descriptor Field Length
$1 \leq N \leq 64$ bytes	Descriptor
3 bytes	Document Number
3 bytes	Document Number

---

* The disk file version of the inverted file does not include the statistical data (first 25 bytes) for the descriptor.

APPENDIX D

CROSS REFERENCE RECORD

LENGTH	CONTENTS
9 bytes	External Document Number
3 bytes	Internal Document Number

APPENDIX E

SEARCH TAPE OUTPUT RECORD FORMAT

POSITION	LENGTH	FIELD DESCRIPTION
0 - 1	2 bytes	Length of Record (217 bytes)
2 - 3	2 bytes	Binary Zeros
4 - 7	4 bytes	Request I.D. from REPORT Card
8	1 byte	Type: C'0' for Searchable C'1' for Text
9	1 byte	Unused
10 - 18	9 bytes	External Document Number
19 - 21	3 bytes	File/Text Segment Number
22 - 24	3 bytes	Category/Text Line Number
25 - 88	64 bytes	Descriptor
89 - 104	16 bytes	Descriptor Links
105 - 168	64 bytes	Subdescriptor
169 - 184	16 bytes	Subdescriptor Links
185 - 200	16 bytes	Value
201 - 216	16 bytes	Second Value